UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

SSPOT-VR: A Space Station for Programming Training in Virtual Reality

Gustavo Martins Nunes Avellar

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Gustavo Martins Nunes Avellar

SSPOT-VR: A Space Station for Programming Training in Virtual Reality

Dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in accordance with the requirements of the Computer Science and Computational Mathematics Graduate Program, for the degree of Master in Science – Computer Science and Computational Mathematics. *EXAMINATION BOARD PRESENTATION COPY*

Concentration area: Computer Science and Computational Mathematics

Advisor: Prof.ª Dr.ª Ellen Francine Barbosa

USP – São Carlos
June 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

M386s

Martins Nunes Avellar, Gustavo
SSPOT-VR: A Space Station for Programming Training in Virtual Reality / Gustavo Martins Nunes Avellar; orientadora Ellen Francine Barbosa. -- São Carlos, 2021.
158 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Teaching and learning of programming. 2. Virtual reality. 3. Block-based programming. 4. Storytelling. 5. Mobile learning. I. Barbosa, Ellen Francine, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Gustavo Martins Nunes Avellar

SSPOT-VR: Uma Estação Espacial em Realidade Virtual para Treinamento em Programação

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *EXEMPLAR DE DEFESA*

Área de concentração: Ciências de Computação e Matemática Computacional

Orientadora: Prof.ª Dr.ª Ellen Francine Barbosa

USP – São Carlos
Junho de 2021

ACKNOWLEDGEMENTS

Throughout the pursuing of this Master's degree I have received a great deal of support and assistance. Therefore, I would like to acknowledge those who helped me directly or indirectly.

I would like to express my deepest appreciation to my advisor, Ellen Francine Barbosa, for her confidence, patience, and friendship. Your expertise is invaluable and you really inspire me. Thank you for everything.

I would also like to extend my deepest gratitude to my family and friends. None of this would have been possible without their support. Your love motivates me to continue everyday. I rely all of my life on my family and funny friends.

The completion of my dissertation would also not been possible without the support and nurturing of my research group brothers and sisters. Thank you for all the help. You also inspire me and made the difference when I got to ICMC-USP.

I also gratefully acknowledge the assistance of colleagues and professors of the *Laboratório de Engenharia de Software* (LabES) and the *Laboratório de Computação Aplicada à Educação e Tecnologia Social Avançada* (CAEd) of ICMC-USP.

Many thanks to the qualified faculty, infrastructure, and staff of the *Instituto de Ciências Matemáticas e de Computação* (ICMC) of the *Universidade de São Paulo* (USP) for providing me a high level graduate education.

I am also grateful to *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil* (CAPES) - Finance Code 001/*Procad* 071/2013, CNPq (134045/2018-1), and FAPESP.

Lastly, I express my sincere gratitude to all those who have contributed, directly or indirectly, to the development of this work.

“Education is the most powerful weapon which you can use to change the world.”
(Nelson Mandela)

Abstract

Technology and computing have become ubiquitous in our lives, both are commonly required to access knowledge, to research and exchange information, to communicate with people and machines, to work, and so forth. The advance and dissemination of computers and mobile devices have also been affecting the world under different aspects: economic, scientific, technological, social, and cultural. In this regard, it is essential to master the fundamentals of technology and computing, including programming skills, to be able to solve problems from any area, to better understand the world we live in, and to act critically as citizens of the 21st century. As programming becomes a fundamental subject in several areas, and a desirable competence in different sectors of society, many teaching and learning approaches have been developed to students and instructors prevail over this complex task. Traditionally, lists of exercises and text-based programming environments are used as support. However, these methods are commonly considered tedious and demotivating, as well as they do not provide many visual stimuli, and can be unsuitable, considering the students' ages. An alternative to text-based programming environments are visual programming languages, such as block-based programming (BBP), that offer increased visual stimulation and ease of use. Improving the teaching and learning resources available to instructors and students may only serve to empower them, as it is possible now to learn using mobile devices (m-learning) as well as experience immersive worlds through virtual reality (VR) apps. Current educational VR apps focused on the teaching and learning of programming do not adopt mobile devices, as well as hardly often use the benefits of immersive VR. Despite the number of researches related to m-learning, VR, and BBP in the teaching and learning of programming, there is a lack of studies in the literature that brings these topics together in the same application. In this Master's research, we propose SSPOT-VR (space station for programming training in virtual reality), a mobile immersive VR block-based programming application for supporting the teaching and learning of programming to students from primary and secondary public and private schools. It integrates methods for the teaching and learning of programming based on educational guidelines and the simulated experience of a digitally created world. SSPOT-VR was created to be experienced using low-cost head-mounted displays (HMDs), although the application can also be experienced directly on the digital screen of mobile devices. Additionally, we aim at connecting students with our application using elements of storytelling and BBP, to engage students in the virtual world atmosphere and simplify interaction. Studies conducted with SSPOT-VR to evaluate its user acceptance and usability were conducted. Obtained results provided evidence that users would accept SSPOT-VR for usage in a real teaching and learning environment, as well as demonstrate its adequate level of usability.

Keywords: Teaching and learning of programming, virtual reality, block-based programming, storytelling, mobile learning.

Resumo

Tecnologia e Computação têm se tornado ubíquas, sendo necessárias para buscar conhecimento, pesquisar e trocar informações, se comunicar com pessoas e máquinas, trabalhar e muito mais. O avanço e a disseminação de computadores e dispositivos móveis também estão afetando o mundo: econômico, científico, tecnológico, social e cultural, fazendo com que seja essencial dominar fundamentos de tecnologia e Computação, incluindo programação, para resolver problemas de qualquer área, melhor entender o mundo e atuar criticamente no século XXI. Com a programação se tornando fundamental em diversas áreas e sendo uma competência desejável em diferentes setores da sociedade, diferentes abordagens de ensino e aprendizagem têm sido desenvolvidas para que os estudantes e os instrutores prevaleçam sobre esta complexa tarefa. Tradicionalmente, listas de exercícios e ambientes de programação baseada em texto são utilizados. No entanto, estes métodos podem ser tediosos e desmotivadores, da mesma forma que fornecem baixo nível de estimulação visual, podendo ser até inadequados, considerando a idade dos estudantes. Uma alternativa são as linguagens visuais de programação, como a programação baseada em blocos (BBP), que oferece alto estímulo visual e facilidade de uso. Melhorar os recursos de ensino e aprendizagem disponíveis é importante para empoderar instrutores e alunos, visto que atualmente é possível aprender utilizando dispositivos móveis (m-learning) e experienciar mundos imersivos através da Realidade Virtual (RV). Os atuais apps de RV, focados no ensino e aprendizagem de programação, não adotam dispositivos móveis, além de raramente aproveitarem os benefícios da imersão. Apesar da quantidade de pesquisas relacionadas a m-learning, RV e BBP, há lacunas para pesquisas que reúnem estes tópicos na mesma aplicação. Neste trabalho de Mestrado, é proposto SSPOT-VR (estação espacial em Realidade Virtual para treinamento em programação), uma aplicação móvel de RV imersiva e programação baseada em blocos, criada para apoiar o ensino e aprendizagem de programação para estudantes Brasileiros da educação básica. O aplicativo integra métodos para o ensino e aprendizagem de programação, baseados em diretrizes educacionais, com a experiência estar em um mundo criado digitalmente. O SSPOT-VR foi criado para ser experienciado utilizando um visualizador de RV de baixo custo (óculos de RV), mas também pode ser experienciado diretamente na tela de dispositivos móveis. Adicionalmente, a aplicação utiliza storytelling e BBP para se conectar com os estudantes, para que os estudantes adentrem a atmosfera da aplicação e também para simplificar a interação com o mundo virtual. Foram conduzidos estudos empíricos com o SSPOT-VR para avaliar sua aceitação de uso e usabilidade. Os resultados providenciam evidências que os usuários adotariam o SSPOT-VR para utilização em um ambiente real de ensino-aprendizagem, bem como demonstram os níveis adequados de usabilidade da aplicação.

Palavras-chave: Ensino e aprendizagem de programação, realidade virtual, programação baseada em blocos, storytelling, aprendizagem móvel.

LIST OF FIGURES

LIST OF CHARTS

LIST OF TABLES

CONTENTS

CHAPTER

1

INTRODUCTION

Several recent studies have reported the relevance of programming skills for today's and for future professionals (JOHNSON *et al.*, 2016; ADAMS *et al.*, 2017; ADAMS *et al.*, 2018). Programming skills have become a commonly required asset for most college and university students (LUXTON-REILLY *et al.*, 2018). It has also become a typical subject among teenagers and children from around the world (MASSO; GRACE, 2011; OH *et al.*, 2013; KALELIOGLU, 2015). The effort of different countries towards the teaching and learning of programming are noteworthy since many of the jobs that will exist in 2030 may not have been invented yet. Hence the need to prepare students and retrain the current professionals with the set of skills required by the new job positions and life in the new age, using the potential of modern technologies in favor of the transformation and development of education and training (UNESCO, 2018).

Since the 1970s, programming skills have been considered as one of the foundations for building knowledge, in particular with respect to the development of powerful ideas and procedural knowledge. This is possible due to that programming skills strengthen the abstraction aptitude and the interaction between our mind and the real world. Students should as soon as possible be prepared to think in cooperation with machines and also to reflect about their own thought process (VALENTE, 2016; PAPERT, 1980). Programming skills contribute to the education of 21st century citizens by enhancing the understanding of the digital world, increasing the capacity for learning and problem solving through the usage of new forms of expression and thought (SBC, 2019; RAABE; BRACKMANN; CAMPOS, 2018).

Programming skills also serve as a support instrument for Brazil's common national curriculum (*Base Nacional Comum Curricular* - BNCC) for primary and secondary education, directly supporting some of the established general skills. The general competence number 1 is about valuing knowledge built in the real and digital world. Number 2 highlights the importance of awakening in students skills for solving problems and developing solutions, including technological solutions. Competence 4 highlights the development of a better understanding of the concept of language and its usage, including the use of programming languages. Finally, the

general competence number 5 elucidates the need for students to use information and communications technology (ICT). With the use of ICTs students must assume the role of active and creative learners and not only usual users of technology (BRASIL, 2018).

Problem solving is essentially associated with abstraction. Given a problem, we need to build an abstract model of reality involving only aspects relevant to the problem. Computing adopts Mathematics to build process models. These models are called algorithms and can be described at different levels of abstraction. Computing provides tools to describe algorithms and supports the process of building and analyzing solutions. The ability to systematize problem solving, represent and analyze solutions through algorithms is also part of programming skills (SBC, 2019).

Considering the largest part of the programming activity as a mental process, i.e., understanding and interpreting the problem and solution, it is possible to affirm that programming is a complex mental activity, requiring both technical knowledge and abstraction capabilities (MINELLI; MOCCI; LANZA, 2015). Consequently, programming skills can develop and enhance other sets of skills, such as attention to detail, strategic thinking, project planning, creativity, and so forth (ROBINS; ROUNTREE; ROUNTREE, 2003; KALELIOGLU, 2015). The process of learning programming is not only restricted to programming theoretical concepts, it is also necessary to acquire and practice construction, comprehension, and analysis of algorithms (KOLLMANSBERGER, 2010; VRACHNOS; JIMOYIANNIS, 2014).

Many researches about the teaching and learning of programming have been adopting different approaches and technologies as supporting mechanisms: (i) block-based programming (BBP), as proposed in the studies conducted by Grover and Basu (2017) and by Bau (2015); (ii) mobile learning (m-learning) applications, as proposed by Tillmann *et al.* (2012) and Jordine, Liang and Ihler (2014); and (iii) different setups of VR and AR as proposed by Chandramouli, Zahraee and Winer (2014), Vincur *et al.* (2017), and Magnenat *et al.* (2015). These studies will be better presented throughout this work.

Even though there are several software applications available to be used as supporting mechanisms to the teaching and learning of programming, as identified by Marcolino and Barbosa (2015), most of the solutions adopted by instructors are still focused on lists of programming exercises and text-based programming environments. Nonetheless the problems and difficulties associated with programming learning still persist. As described in different studies, the causes of the problems and difficulties are extensive and can be related to (SOUZA; BATISTA; BARBOSA, 2016; SENTANCE; CSIZMADIA, 2017; CHEAH, 2020; NOURI *et al.*, 2020; FRANCISCO, 2021): (i) the lack of a universal definition of what to learn for part of the educational stages; (ii) students' motivation or ability to problem-solving; (iii) students' competence in learning and applying programming concepts; (iv) instructors adoption of static and traditional teaching materials, and (v) school or university curriculum.

Since many students consider programming a laborious and tedious activity, their negative

perception influences their attitudes toward learning (CORNEY; TEAGUE; THOMAS, 2010; SHI; WHITE, 2013). Supporting the students' feelings, the Cone of Learning, created by Dale (1969), asserts that after two weeks the human brain tends to remember 90% of what students experienced with practice or simulation, as opposed to only 10% of what it is read, 20% of what it is heard, and 30% of what it is seen.

At the current stage of computer graphics and hardware for desktops, laptops, and mobile devices, it is possible to insert users into a virtual environment and create an almost perfect sensation of physical existence of that space (TORI, 2010b). In this Master's research, we focused on applying together the concepts of immersive VR, mobile learning, block-based programming, and storytelling on the teaching and learning of programming for children and teenagers. We rely on the programming teaching guidelines provided by SBC (2019) and Raabe, Brackmann and Campos (2018). According to a mapping study we present in Section 2.4, there is no mobile immersive VR application that applies block-based programming principles and storytelling elements for the teaching and learning of programming concepts to children and teenagers.

Block-based programming (BBP) has become well-known among instructors and students over several applications and platforms, e.g., Scratch, App Inventor, Code.org, Blockly (AIVALOGLOU; HERMANS, 2016). However, a visual programming language refers to many benefits to the learning process, conventional ways of interaction such as computer peripheral devices and digital screens restrict students from improving their engagement with the learning activity. With the current growth in availability of devices compatible with VR and AR, there are possibilities for making BBP more immersive, using sophisticated equipment or through the adoption of mobile devices, that usually contain all the necessary components to execute a VR or AR application (VINCUR *et al.*, 2017; TORI; HOUNSELL, 2018).

Due to the reduced size and cheapening of electronic components, mostly today's mobile devices have features and processing power equivalent or superior than conventional computers (ZAMFIRACHE; OLTEANU; TAPUS, 2013). Such hardware advancements associated with the advent of ubiquitous computing and the increasing adoption of digital learning environments have led to a new learning paradigm, named mobile learning (WEXLER *et al.*, 2008). The m-learning term implicitly means mobile electronic learning and its concept evolution over the years is conceived as both an extent of traditional electronic learning and a reaction to its limitations (TRAXLER, 2009). One of m-learning main characteristics is concerned with the student mobility, specifically that students should be able to engage in educational activities without being in a fixed place, even when the m-learning application is based in any level on VR or AR or other (KUKULSKA-HULME; TRAXLER, 2007; TORI; HOUNSELL, 2018).

As well as block-based programming and mobile learning, virtual reality also contributes to the teaching and learning of programming. Besides inserting users into a fully immersive virtual environment, VR allows the experience and manipulation of situations and objects that a physical presence or interaction would be impossible, expensive, or dangerous. VR enables

us to create or replicate numerous situations of information visualization, problem solving, and manipulation at various levels of detail and abstraction (QUEIROZ; TORI; NASCIMENTO, 2017). Another important attribute of VR in education is that VR is able to produce the sensation of presence, i.e., the feeling of being part of what seems to be a real world. When this happens, people naturally becomes actively engaged in the task that is being performed, increasing levels of motivation and commitment (TORI; HOUNSELL, 2018). For instance, it is possible to virtually insert people inside a program compiler, allowing the visualization of the various states of the program and the interaction with its variables. All these experiences may help to minimize the problems and difficulties in the teaching and learning of programming, as well as they fortify the solutions proposed by several studies in the literature (PITEIRA; COSTA, 2013; SOUZA; BATISTA; BARBOSA, 2016; MARCOLINO; BARBOSA, 2017a; LUXTON-REILLY *et al.*, 2018).

The effective adoption of VR has presented challenges for its popularization due to the dependence on equipment and complexity to create applications. Development tools were usually available only for companies or specialists. As mentioned, nowadays with the technological advancement, several essential components for VR applications have become part of mobile devices. Besides the mobile devices compatibility, after 2014 several cardboard and plastic viewers (VR adapters or head-mounted displays – HMD) appeared. These HMDs hold no electronic parts, can be purchased or built for a low cost, and only need a compatible mobile device to work (CARDOSO *et al.*, 2017).

From a pedagogical and educational point of view, the advantages of interactive 3D applications are that 3D applications help to reduce the impression of complexity, encouraging learning by experience and providing an immersive effect which improves concentration and willingness to solve problems (SEGURA *et al.*, 2020). Moreover in the Hype Cycle for Emerging Technologies of 2017 (GARTNER, 2017), VR is considered one of the emerging technologies for the next 2 to 5 years, i.e., it is about to become a technology for the large commercial use, considering its current availability, robustness, and reliability.

Considering: (i) the importance of programming skills for the education of children and teenagers; (ii) the lack of mobile immersive VR apps focused on the teaching and learning of programming (AVELLAR; BARBOSA, 2019), and (iii) that m-learning applications, BBP, and VR have been separately supporting the teaching and learning of programming (TILLMANN *et al.*, 2012; BAU, 2015; GROVER; BASU, 2017; VINCUR *et al.*, 2017), this Master's research aims to propose, develop, and evaluate a mobile immersive VR application for teaching programming to children and teenagers using BBP principles and storytelling elements. The main objectives are summarized next.

1.1 Research Objectives

According to the context and motivation presented in the previous section, the research question investigated in this Master's work is whether or not students from primary and secondary public and private schools would accept the usage of a mobile immersive VR application to support the learning of programming.

Based on this research question, the main objectives of this Master's work are described as follows:

- **Study the scenario of teaching and learning of programming:** we aim to provide an overview of the teaching and learning of programming research area. The main idea is to characterize the current scenario of this field and investigate problems and solutions associated with the processes of teaching and learning of programming.
- **Study the research area of VR and AR applications for teaching and learning of programming:** we aim to present how VR and AR are being used for supporting the teaching and learning of programming. We aim to particularly inspect immersive VR works, with emphasis on applications for mobile devices that adopt visual programming languages as learning mechanisms.
- **Create a mobile immersive VR application:** we aim to propose and develop a mobile immersive VR application based on block-based programming principles and storytelling elements to support the teaching and learning of programming for children and teenagers from primary and secondary public and private schools.
- **Evaluate the application:** we aim to plan and conduct studies with students from primary and secondary public and private schools to evaluate the acceptance and usage of the mobile immersive VR application. We also aim to evaluate the application usability levels through an heuristic evaluation.

1.2 Outline

In this Chapter we described the context that this Master's research is inserted, as well as the motivations for its accomplishment, in addition to the main objectives of this work.

In Chapter 2 we present an overview of the background information that supports the topics investigated in this work. Initially, we discuss the current scenario of teaching and learning of programming, as well as the idea of block-based programming and storytelling. Then, theory and concepts associated with mobile learning, virtual and augmented reality are addressed. We also provide a mapping study which regards to VR and AR in the teaching and learning of programming.

In Chapter 3 we present the SSPOT-VR application, a Space Station for Programming Training in Virtual Reality. We present all the information related to the operation and usage of the application and details of its educational and technical aspects.

In Chapter 4 we present two SSPOT-VR evaluations. The first evaluation, carried out with children and teenagers, aimed to verify factors related to user acceptance and use. The second, made by experts, aimed to assess user experience and usability of SSPOT-VR.

In Chapter 6 we conclude this dissertation, reviewing contributions, summarizing limitations, and presenting perspectives of future research.

CHAPTER

2

BACKGROUND

In this Chapter we provide an overview of the subjects that underlie the research developed in this work. In Section 2.1, we present an overview of the contemporary scenario of the teaching and learning of programming, including the current problems and challenges, and efforts towards solutions. In Section 2.2, we introduce the fundamental concepts related to virtual reality: definitions, history, classifications, and characteristics. In Section 2.3 we introduce the main concepts of augmented reality, along with its definitions, comparison with Virtual Reality, and classifications. In Section 2.4, we characterize the state-of-the-art of virtual and augmented reality applications for supporting the teaching and learning of programming according to the results of a systematic mapping study of the literature. Final remarks are given in Section 2.5.

2.1 Teaching and Learning of Programming

Programming has become a common discipline in most college and university courses (ACM; SOCIETY, 2005; LUXTON-REILLY *et al.*, 2018; ACM; SOCIETY, 2020). As well as in secondary, primary, and even early childhood education, whereas initiatives such as STEM (science, technology, engineering, and mathematics) have been achieving worldwide acknowledgment (RAABE; BRACKMANN; CAMPOS, 2018; SBC, 2019; CAMPOS; DIAS, 2020). Programming has also become an important competence for everyone in the present day. It is imperative to have core skills in programming to live in today's increasingly digital society (MASSO; GRACE, 2011; CHANDRAMOULI; HEFFRON, 2015).

Programming skills are fundamental for living and acting in today's society since they support the development of essential skills for students and professionals, such as abstraction, inquiry, problem solving, flexibility, and so forth. Regardless of area of study or occupation, these individual characteristics can be explored to make one understands how to create using digital technologies and not only benefit from their plain usage (VALENTE, 2016; ROBINS; ROUNTREE; ROUNTREE, 2003). Wing (2011) states that programming skills, as well as

reading, writing and arithmetic, are essential to develop the analytical capacity in children and teenagers. Programming skills also reinforce attitudes, such as confidence in dealing with complexity, persistence while working with difficult problems, tolerance of ambiguity, and the ability to deal with open problems (CSTA; ISTE, 2011). Therefore, programming is not a single skill, but a set of resources, competences, and experiences associated with a complex cognitive activity (FRANCISCO, 2021).

Proficient programmers are efficient in recognizing, using, and adapting patterns or schemes, as well as they are more able and accurate to understand a wide range of examples from different sources of knowledge and apply effective strategies in order to solve a problem(ROBINS; ROUNTREE; ROUNTREE, 2003). The programming activity can also stimulates the development of collaboration, coordination, and communication, for both students and professionals. In this regard, a common approach is pair programming, in which two people work together to solve a problem (MARCOLINO; BARBOSA, 2017a). Programming skills also allow students to think together with computers and mobile devices, in addition to thinking about their own thinking (PAPERT, 1980).

This reasoning has led to the development of several public and private initiatives around the world to support the teaching and learning of programming for children and teenagers, including changes in the primary and secondary education curriculum of several countries, in which programming topics are being introduced even in the early years of primary school. In the United States, under the statement that everyone should learn how to code [1,2], many educational initiatives have been originated to support the teaching and learning of programming. The Next Generation Science Standards (NGSS) are content standards to help instructors and students from primary and secondary school to teach and learn the different concepts of Science and Engineering, including programming. The standards allow instructors to design classroom learning experiences that stimulate students' interests in Science and prepares them for college, careers, and citizenship [3].

In 2015, approximately 30% of all U.S. students have accounts on Code.org [4], a nonprofit that started in 2013 and is focused on expanding the access to Computer Science (CS) of women and underrepresented minorities from primary and secondary schools of the US. Code.org adopts different approaches to teach programming, including the usage of visual programming languages (VPL), such as block-based programming (BBP). Their vision is that every student in every school should have the opportunity to learn programming (KALELIOGLU, 2015).

Code.org also organizes the annual Hour of Code[5] campaign. The Hour of Code started as an one hour event to present Computer Science concepts and demystify the activity of

[1] https://goo.gl/sCD48L
[2] https://tinyurl.com/4rb6r59x
[3] https://www.nextgenscience.org/
[4] https://code.org/
[5] https://hourofcode.com

programming, to demonstrate that anybody can learn how to do it, and to broaden participation in the field of Computer Science. A related Brazilian initiative, entitled *Desafio do Código*[6], aims to stimulate the learning of programming, Mathematics, and English through fun and online challenges. The purpose of *Desafio do Código* is to disseminate programming skills, especially to children and adults that had any lack of education or live in precarious social conditions.

In Europe, the report published by Balanskat and Engelhardt (2015) describes the current situation of the teaching and learning of programming in the primary and secondary education of 20 European countries. The curriculum of primary and secondary education of 13 of these countries currently contain mandatory programming topics. In England, since the implementation of a new curriculum for the United Kingdom in 2014, several technology and Computer Science topics are mandatory in primary and secondary schools, addressing contents from three different thematic units: Computer Science, information technology, and digital literacy.

The U.K. curriculum aims to create conditions to elucidate how digital technologies work, their impacts and relations with society, and the diversity of usage in different contexts and situations (RAABE; BRACKMANN; CAMPOS, 2018; VALENTE, 2016). The U.K. government implemented these changes mainly due to the belief that programming is not only coding. In many cases it helps with Mathematics, English, and strategic thinking. Additionally, there were complaints of technological industries related to the need of more qualified fresh graduates in England (DUNCAN; BELL TIM ANDTANIMOTO, 2014).

In Australia, the integration of computing into primary and secondary education occurred in part due to the natural growth of instructors and schools in their adoption trajectories of Information and Communication Technologies (ICTs). After a while using ICTs to support the teaching and learning of different subjects, instructors realized that focusing only on the ICTs use was not enough in comparison with the educational potential of these technologies. The computing topics approached are divided into two units: design and technologies and digital technologies (RAABE; BRACKMANN; CAMPOS, 2018). The Australian government has also justified the insertion of these computing units in the curricula with projections that show a lack of qualified software developers. If this situation is not changed, it may result in development problems for the country in some years[7]. The Australian curriculum is based on successful experiences that took place in the U.S., the Code.org and Hour of Code (DUNCAN; BELL TIM ANDTANIMOTO, 2014).

The reductions in the cost of ICTs contributed to their adoption in education, allowing the usage of different resources to support formal education and the appearance of new learning methods, such as electronic learning (e-learning) and mobile learning (m-learning). Several new hardware and new technologies have also been adopted, such as mobile and wearable devices, Virtual (VR), Augmented (AR), and Mixed Reality (MR), artificial intelligence (AI),

[6] http://desafiodocodigo.com.br/
[7] https://goo.gl/Teo6EX

microcontrollers, Internet of Things (IoT), and so on (CRAIG *et al.*, 2012; RUBENS; KAPLAN; OKAMOTO, 2014).

There are also initiatives with other aims, such in Estonia[8], where the objective is the intellectual enhancement. The government believe that people that use technology, computers, and the web become smarter. It was launched a national scheme to teach basic programming to school children and teenagers from the age of 7 to 19. The idea was not to create a mass of software developers for the future, but prepare people to have digital literacy in order to create a smarter association with technology, computers, the web, and life (DUNCAN; BELL TIM ANDTANIMOTO, 2014; VALENTE, 2016). In Italy, computing topics are being integrated with the regular subjects of primary and secondary education primarily for exploring critical attitudes in the usage of digital technologies for work and life. The topics also focused on the usage of computers to retrieve, evaluate, maintain, produce, present, and share information through the Internet. The development of programming skills is up to the instructors, who teach fundamental concepts of programming through robotics and visual programming languages (MANNILA *et al.*, 2014).

In Brazil, the common national curriculum (*Base Nacional Curricular Comum*, BNCC) specify several general skills for students to develop throughout primary and secondary education. Among the competencies, there is a need to study computing concepts as of the early years of primary education, including programming concepts. The goal is to empower students, enabling them to impact their own lives and their community using technology and Computer Science (SBC, 2019). BNCC is a normative document that ensures the learning rights of Brazilian students and defines the set of skills that all students must develop throughout the stages of primary and secondary education (BRASIL, 2018).

The SBC (2019) constructed computing curricular guidelines to assist the teaching and learning of programming concepts in the primary and secondary education, in addition to many topics related to Computer Science and digital technologies. The objective of SBC is to promote access to information and culture through information technology, to promote digital inclusion, and to encourage research in computing education in Brazil. The document establishes a starting point for the inclusion of computing concepts in primary and secondary education. The computing curricular guidelines define different topics and skills in three thematic units: computational thinking, digital culture, and digital world. SBC is a non-profit scientific society that brings together students, professors, professionals, and researchers from Computer Science and Informatics.

The *Centro de Inovação para a Educação Brasileira* (CIEB) has also constructed a reference curriculum in technology and computing for primary and secondary education. CIEB is a non-profit organization which mission is to promote innovation in public education in Brazil. The curricula objective is to support schools with a quality material, that is practical and flexible,

[8] https://goo.gl/4kC9Ya

to make it possible to transversely work the computing topics or create specific subjects. The document provides a set of skills to be acquired, pedagogical practices, assessment methods, and reference materials, in three thematic units: digital culture, digital technology, and computational thinking (RAABE; BRACKMANN; CAMPOS, 2018). In addition to the SBC guidelines and the CIEB reference curriculum, the municipal curriculum of São Paulo schools also approaches the learning of technology and computing. It requires the schools to address programming concepts, ICTs usage, and topics on digital literacy ((SP), 2017).

Another Brazilian initiative is the Technovation Summer School for Girls [9], annually offered since 2019 by GRACE (*Grupo de Alunas nas Ciências Exatas*) of *Instituto de Ciências Matemáticas e de Computação* (ICMC) from the *Universidade de São Paulo* (USP). The event helps to prepare girls aged 10 to 18 years old for the Technovation Challenge[10], an international competition of entrepreneurship and technology which goal is to solve real-world problems through technology (TRIDICO *et al.*, 2018; RIDEL *et al.*, 2018). There are other public Brazilian initiatives: the programming literacy program of Ayrton Senna institute[11]; the hackers' school from *Passo Fundo*, *Rio Grande do Sul*, Brazil[12]; the *Programaê!* program of Lemann and Vivo foundations [13], and the Go Code program, of Maurício Sirotsky Sobrinho foundation [14] (VALENTE, 2016).

Private initiatives, such as Ctrl + Play School of Programming and Robotics[15], Yadaa[16], and Happy Code[17] focuses on the teaching and learning of programming and robotics for children and teenagers through playful methods. The schools emphasize that children and teenagers usually spend hours in front of mobile devices, hence the usage of these devices should also serve to something meaningful to their lives, such as learning programming through playful and educational tools. With the content approached in the school courses, it is possible to create games, applications, websites, and robots.

The emphasis on programming learning can also occur to prepare students specifically for work. This learning takes place through several specific disciplines that addresses programming concepts together with the regular subjects of secondary education. In Brazil and Europe, schools with professional education integrated with regular education operate accordingly (MANNILA *et al.*, 2014; BALANSKAT; ENGELHARDT, 2015).

Even though the increasing significance of programming skills among different audiences and its global notoriety, there is still cause for concern about the problems and challenges

[9] http://grace.icmc.usp.br/TechSchool/
[10] https://technovationchallenge.org/
[11] https://institutoayrtonsenna.org.br/pt-br/como-atuamos/letramento-em-programacao.html
[12] https://www.upf.br/extensao/projetos-programa/projeto-escola-de-hackers
[13] http://programae.org.br/
[14] http://www.fmss.org.br/go-code/
[15] https://www.ctrlplay.com.br/
[16] https://www.yadaa.com.br/
[17] https://www.happycodeschool.com/

associated with the teaching and learning of programming and the lack of software development professionals (CUSHING; GANTZ, 2013; JOHNSON *et al.*, 2016).

Although the popularization of computers and mobile devices has increased the possibility of using digital technologies in primary and secondary education, they have been slightly used to stimulate the development of programming skills in students, providing only minor contributions to the understanding of the functioning of computers and computational concepts (VALENTE, 2016).

Besides that the teaching and learning of programming is considered a complex task and programming subjects and courses generally have high rates of dropout (ROBINS; ROUNTREE; ROUNTREE, 2003; PITEIRA; HADDAD, 2011; ALLEN *et al.*, 2018). Thus it is imperative that instructors adopt suitable methods for their particular public, in order to keep students' motivation and commitment while learning. Considering that children and teenagers are commonly held back by traditional teaching and learning methods, the method should be primarily age-appropriate, and introduced as a form of entertainment in association with other formative processes, such as technology, art, or reading (SYROCKA, 2017). As well as the adoption of ICTs and novel learning methods, the benefits and challenges related to the teaching and learning of programming have driven several scientific researches to have a better grasp of the research area and to assist students and instructors to prevail over the complex task of learning and teaching programming.

Next, we address the main problems, challenges, and solutions associated with the teaching and learning of programming.

2.1.1 Problems, Challenges and Solutions

Despite the significance of programming skills for today's students and professionals and several initiatives and research to broaden participation in the field of Computer Science, programming is still complex to learn and presents several issues during its learning process. Reports from instructors of programming and results from empirical studies suggest that the teaching of programming has created significant problems and challenges for both primary and secondary school and university students (SLEEMAN, 1986). The problems, difficulties, and challenges associated with the teaching and learning of programming are extensive and can be related to multiple parameters. These problems have been widely discussed (ARANHA, 2015; QIAN; LEHMAN, 2017; LUXTON-REILLY *et al.*, 2018; CHAKRAVERTY; CHAKRABORTY, 2020; CHEAH, 2020).

Souza, Batista and Barbosa (2016) mapped several problems and difficulties in the teaching and learning of programming research area. According to the authors, the problems most frequently addressed by primary studies include: (i) difficulty in learning programming concepts; (ii) difficulty in applying programming concepts; (iii) motivation issues, and (iv) instructors issues. The authors also stated that most of the studies are designed to aid not only a

single difficulty or problem, but a group of them, such as the difficulties of learning and applying programming concepts; and the difficulty of learning programming concepts and motivation issues.

Instructors of programming subjects from Brazilian universities confirmed the presence of these problems and difficulties in their daily and also added some of their personal problems: (i) students are not well qualified to code, their knowledge in Mathematics and logic are insufficient; (ii) students are not engaged, they prefer to use smartphones and the internet instead of learning; (iii) students copy assignments (plagiarism); (iv) programming environments and languages provide little visual stimulation; (v) students do not like to work in groups, and (vi) programming subjects are often inflexible and somehow outdated (MARCOLINO; BARBOSA, 2017a).

According to the review conducted by Sentance and Csizmadia (2017), the most common challenges experienced while teaching Computing topics, including programming, are: (i) teachers' own subject knowledge; (ii) students' difficulty in understanding concepts; (iii) technical problems in school; (iv) differentiation to meet different levels of ability, and (v) students willingness or ability to problem solve. From a Brazilian viewpoint, Santana, Araujo and Bittencourt (2019) add the following challenges: (i) lack of adequate infrastructure in schools; (ii) problems and difficulties in integrating Computing with the current curricula, and (iii) lack of instructors training (SANTANA; ARAUJO; BITTENCOURT, 2019).

In general, students do not know how to decompose the initial task in ordered steps, such as a flowchart, pseudo-code, or algorithm. This is mainly due to their lack of problem-solving skills (AISSA *et al.*, 2020). Programming requires one to simultaneously apply several higher-order cognitive skills to develop solutions to real world problems and implement them in a computational environment (FRANCISCO, 2021). Specify a detailed plan that can be carried out is not trivial: many people are unable to say how they perform certain tasks (SLEEMAN, 1986).

Students also face difficulties to understand and apply the basics concepts of programming, that allows them to develop computational thinking and will give them the ability to design algorithms to solve specific problems. Ensuring the learning of the basics programming concepts is important to prevent students from developing a negative attitude towards programming (SERALIDOU; DOULIGERIS, 2021). In addition to understanding the basics of programming, students commonly have difficulties understanding complex programming concepts such as data structures, arrays, pointers, and so on (PITEIRA; COSTA, 2013). On the other hand, many students understand the programming concepts, but have difficulties using them while creating algorithms. Student's deep understanding is not entirely memorization, it also involves application of theoretical knowledge (PANWONG; KEMAVUTHANON, 2014).

Students' negative perception influences their attitudes toward learning programming. It spoils the intrinsic motivation of students to continue learning for success (QIAN; LEHMAN, 2017). The motivation and commitment issues could be associated with the programming

environments and languages, that usually provide little visual stimulation and demand high level of abstraction. Students also often give up solving a problem if they do not quickly find a possible solution (AISSA *et al.*, 2020). These issues are strengthened by the traditional and static teaching materials and lack of practical activities. It is also worth mentioning that the teaching and learning methodologies are seldom focused on the diversity of learners and their individual difficulties (SOUZA; BATISTA; BARBOSA, 2016; MARCOLINO; BARBOSA, 2017a).

In contrast with the results Souza, Batista and Barbosa (2016) obtained, the motivation issues was ranked as the least important in the study conducted by Marcolino and Barbosa (2017a), indicating that either the instructors consider their students highly motivated to learn programming, or they are unable to asses the students' motivation.

Therefore, there are also problems concerning the effectiveness of the teaching material and pedagogic approaches. Many instructors use only traditional teaching materials and methods which are either not suitable or tedious. A quite common approach is the application of lists of programming exercises (MARCOLINO; BARBOSA, 2017a). Then, it is essential to adopt modern and interactive tools and approaches to maintain student's engagement and to allow them to take an active role within the learning process (AISSA *et al.*, 2020). Static materials such as printed books are not as effective as dynamic tools, such as simple animations with interactive examples of programming concepts (CHEAH, 2020).

In the case of adopting a programming language, it should have straightforward commands, usage cannot be complex, and it has to be self-explanatory (CHEAH, 2020). Although instructors should be rather concentrated in fostering the development of problem-solving skills than on the learning of a programming language (AISSA *et al.*, 2020), when adopting programming languages, it is necessary to clearly explain why and how the algorithms work, the objectives of any given algorithm, strategies for decomposing tasks, rules to create algorithms, and design strategies (SLEEMAN, 1986).

At the same time, the insufficient universal definitions of what to teach in primary and secondary education regarding programming is also a challenge. How to teach is pointless without knowing what to teach (NOURI *et al.*, 2020). In Brazil, there was a lack of curricular guidelines and reference curriculum for the teaching and learning of programming in the primary and secondary education. However, in the last years the Raabe, Brackmann and Campos (2018), Campos and Dias (2020), and SBC (2019) have developed documents with this purpose. In addition to the need to define curricular references, it is also necessary to create appropriate teaching materials to Brazilian students. Santana, Araujo and Bittencourt (2019) created a Computing digital book for each year of secondary school. Although it is a textbook, its goal is to integrate concepts from Computer Science, encourage the use of ICTs, and help in the development of programming skills.

The studies selected by Souza, Batista and Barbosa (2016) also proposed solutions, which are mostly related to: (i) usage of enhanced visualization methods; (ii) adoption of

serious games; (iii) usage of educational environments; (iv) collaborative programming, and (iv) scaffolding. Authors observed that the categories of problems and solutions that are more often related are difficulty in learning programming concepts and visualization supporting tools. These tools are also frequently related to the application of programming concepts. Other relations between problems and solutions are: (i) serious games to support the learning of programming concepts and increase motivation; and the use of educational environments to help the learning of programming concepts and application of programming concepts (SOUZA; BATISTA; BARBOSA, 2016).

Similar results are described by Chakraverty and Chakraborty (2020). The categories they formed to represent the solutions identified in their review are as follows: (i) visual programming languages; (ii) game-based learning; (iii) collaborative programming; (iv) robot programming, and (v) assessment and personalized feedback platforms.

In the study conducted by Marcolino and Barbosa (2017a) instructors informed what they adopt in terms of tools and methods to assist in the teaching and learning of programming: (i) adoption of easy programming languages, such as Python, which has fewer complex structures, weak variable typing, and less reserved words; (ii) usage of visual programming languages and other programming-aid applications, such as Scratch[18], Greenfoot[19], and VisualG[20]; (iii) adoption of different teaching methods and materials, such as programming competitions and problem-based learning (PBL); and (iv) usage of ICTs and digital media.

The instructors' perspective of solutions is also presented by Sentance and Csizmadia (2017): (i) learning away from electronic devices (unplugged-style); (ii) collaborative programming; (iii) contextualisation of learning, and (iv) scaffolding programming tasks.

Support tools are usually created especially for novice students, their goal is to make the learning process more effective through the adoption of different teaching materials, pedagogic approaches and/or technology (FRANCISCO, 2021). Besides that, according to Santos *et al.* (2020), most studies that present tools or approaches to support the teaching and learning of programming are for undergraduate students enrolled in introductory programming subjects. As stated by Pears *et al.* (2007), it is possible to classify an educational application according to its strategy, such as: (i) visualization tools; (ii) automatic evaluation tools; (iii) programming support tools; (iv) microworlds; and (v) others tools, that approaches strategies not included in the previous categories.

Among the visualization strategies, it is emphasized those that adopt simplified programming methods, such as the use of BBP. The use of visualization strategies mainly supports the learning of programming concepts. Since a visual appealing activity is the intermediary between the student and the programming content, students' motivation is also enhanced. There

[18] https://scratch.mit.edu/
[19] https://www.greenfoot.org
[20] http://visualg3.com.br/

are also initiatives that promote the usage and creation of storytelling animations and those that use metaphors for the representation of commands and functions (MARCOLINO; BARBOSA, 2015). The use of metaphors and storytelling is classified as microworlds, that designates learning environments in which it is possible to explore, discover, and simulate real world events (JONASSEN, 2007). Meerbaum-Salant, Armoni and Ben-Ari (2010) emphasize that visual programming languages ease coding and turn programming to an enjoyable and not frighten activity.

In addition, microworlds and mobile devices have being capable to increase the sense of closeness perceived by students. Even though it is not yet possible to fully replace formal education, such as the experience of physically manipulate an object, interactive and mobile technologies can minimize the distance effects that take place during the learning process (TORI; HOUNSELL, 2018). Mobile and ubiquitous technologies have been described as a potential facilitator which enables learners to learn across contexts with access to learning resources in anywhere and at any time (HWANG; LI; LAI, 2020). Today, the resources and techniques initially designed for electronic education are being applied in formal education, instructors and students are discovering that virtual resources can be a major support for face-to-face activities, such as in the scenario that are possible to simulate the physical presence of an instructor using VR (TORI, 2010a). Many researchers argue that these environments have a positive effect on the motivation for programming and that through exploring and solving problems in realistic environments students are more engaged in the activity (SERALIDOU; DOULIGERIS, 2021).

The usage of VR presents a step further to a even more appealing and immersive experience. The educational area has been significantly served by VR since this technology has become popular and affordable nowadays, making it possible to learn through virtual laboratories, representation and manipulation of virtual objects, remote classes, participation in virtual events, and so on (TORI; HOUNSELL, 2018).

The association of BBP, mobile devices, and VR may support the teaching and learning of programming by providing simplified programming methods, mobility, and immersion to the students during the learning process, as well as helping them to develop a positive attitude toward learning programming. Particularly in relation to BBP and VR, their usage in conjunction enable the natural visual stimulation of BBP to be more immersive. Hence, BBP may be more suitable to VR environments than approaches using fully textual programming languages. In addition, BBP is considered an useful support mechanism to novices while learning programming.

Next, we provide an overview about block-based programming.

2.1.2 Block-based programming

Block-based programming (BBP) is a type of visual programming language (VPL) based on a jigsaw metaphor. BBP languages, tools, or environments, enable users to assemble

functional applications using only drag-and-drop interactions (WEINTROP; WILENSKY, 2015). Combining the advantages of component-based programming and end-user programming, BBP allows the integration of several pieces of code in a form of blocks to develop applications in a short period of time, within estimated cost, and without any or with little programming skills needed (MOHAMAD *et al.*, 2011).

Component-based programming is based on code organizing principles to create software components with a certain set of features that can be integrated into larger pieces of software, providing flexibility and reusability to the development process as well as minimizing the efforts to create new features or replace old features (MOHAMAD *et al.*, 2011). According to Bachmann *et al.* (2000), component-based development (CBD) produces applications by merging prefabricated components with pieces of code that provide bonding between the components and their new functionality. On the other hand, end-user programming is an approach to allow people who are not professional software developers to produce or modify a software artifact (MOHAMAD *et al.*, 2011).

Although BBP languages are visual languages, it still adopted certain aspects of text-based languages, such as limited text-entry and indentation. This makes BBP more similar to textual programming than other visual programming languages, as the dataflow languages (HERMANS; AIVALOGLOU, 2016). BBP exists since 1986, when Glinert (1986) introduced the BLOX language, which consists of puzzle-like programming commands that are assembled into programs by combining them vertically and horizontally. However, as stated by Hermans and Aivaloglou (2016), only recently BBP has been largely used as a tool for the teaching and learning of programming.

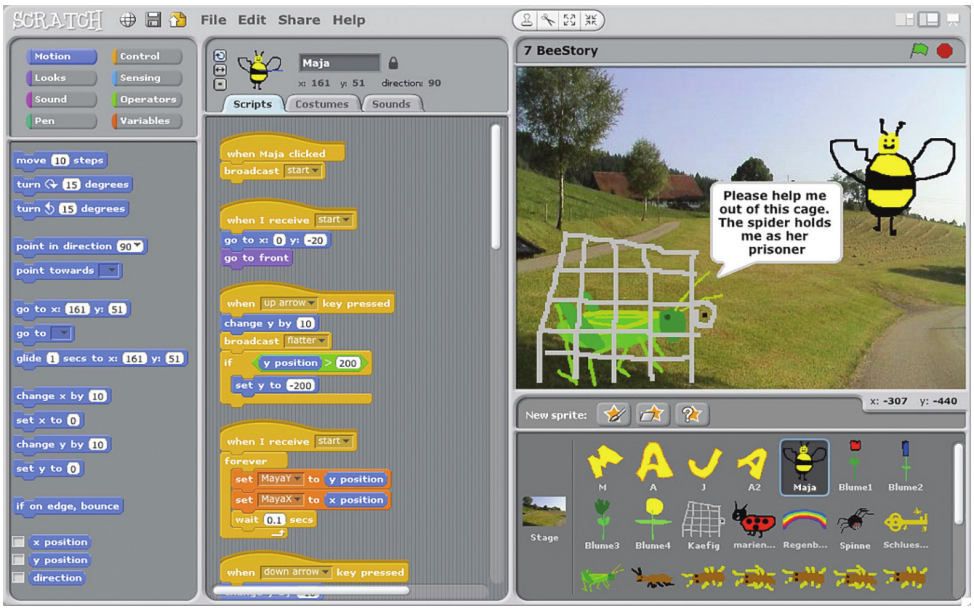
BBP gained notoriety especially with Alice, an open-source drag-and-drop programming environment that enables novice programmers to create animations with 3D models (COOPER; DANN; PAUSCH, 2000). The BBP approach include a number of features that have been identified as especially productive for novice programmers of any age, including support to commands using natural language, logically ordered commands, and the drag-and-drop assembly mechanism. These features help to reduce the memorization that is required in text-based programming and also the general effort, since assembly blocks is easier and faster than typing commands using a keyboard (WEINTROP; WILENSKY, 2015; WEINTROP *et al.*, 2017).

The advent of the BBP paradigm has resulted in several programming tools and environments that have introduced many users to concepts of computing (BAU *et al.*, 2017). Since then, new BBP languages have gained widespread popularity, such as Scratch (RESNICK *et al.*, 2009), App Inventor (WOLBER *et al.*, 2011), and Blockly[21]. Currently, over 100 million students have tried Blockly via Code.org, the Scratch repository hosts over 12 million projects, and the App Inventor holds over 400,000 unique monthly active users who come from 195 countries (HERMANS; AIVALOGLOU, 2016).

[21] https://developers.google.com/blockly/

As explained by Resnick *et al.* (2009), Scratch was one of the first BBP language developed, it was launched by the Massachusetts Institute of Technology (MIT) and is focused on children and teenagers. It is available for usage both as a desktop, web or mobile application and can be used to create games, simulations, and interactive animations (Figure 1). Even though a considerable group of adults also use the BBP language, the target and core audience of Scratch is children and teenagers between the ages of 8 and 16 years old.

Figure 1 – Scratch interface



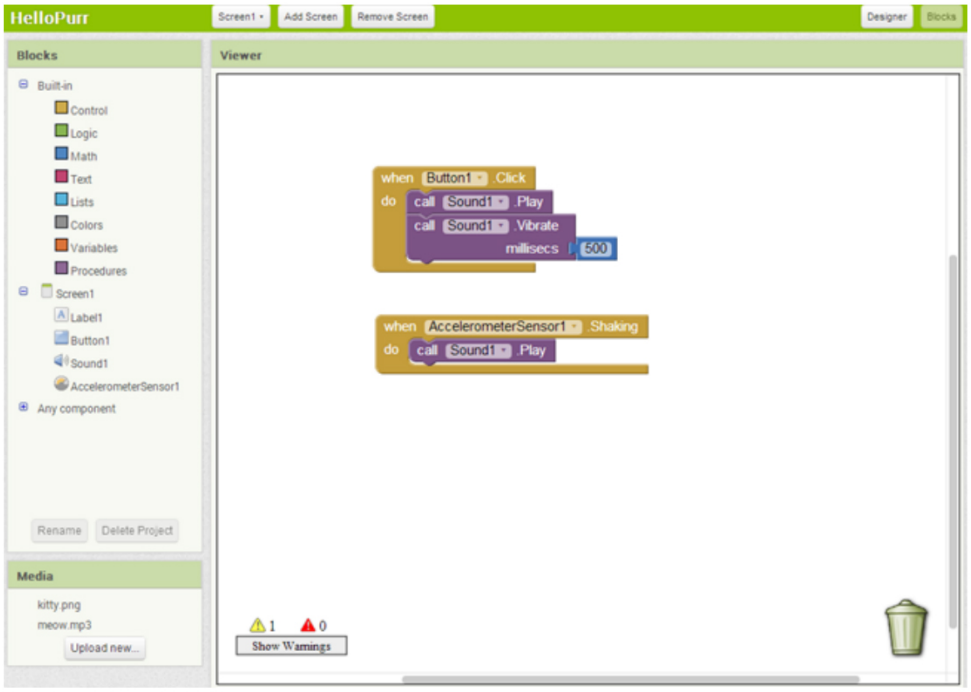
Source: Resnick *et al.* (2009)

Similar to Scratch, the App Inventor is an intuitive BBP web or desktop environment (Figure 2) firstly created by Google and now maintained by MIT, that allows everyone to build functional applications for mobile devices (WOLBER *et al.*, 2011). Scratch and App Inventor were created to make programming easier for everyone, of all ages, backgrounds, and interests. The platforms seek to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation. As the users of Scratch and App Inventor develop and share their projects, they learn important Mathematics and Computing concepts, as well as how to think creatively and systematically, as well as how to work collaboratively, which are all essential skills for the 21st century (RESNICK *et al.*, 2009; WOLBER *et al.*, 2011).

As reported by Weintrop *et al.* (2017), BBP tools are effective to enable novices to write successful programs with none or little prior experience, as well as they can serve as an accessible introduction to the learning of traditional programming languages (WEINTROP *et al.*, 2017). According to Bau *et al.* (2015), the approaches of teaching programming generally adopts one of two methods: (i) they help beginners achieve satisfying results quickly in a way that minimizes frustration; or (ii) they introduce beginners directly to programming environments

used by professionals.

Figure 2 – App Inventor interface



Source: Wolber *et al.* (2011)

In this context, Bau *et al.* (2015) created the Pencil Code, a BBP tool that helps in novices' transition to work with text-based programming languages. The tool was designed to bridge these two types of programming and has been used to help beginners of all ages to create programs in JavaScript and CoffeeScript. The goal of Pencil Code is to build enough confidence in novice programmers to enable them on creating programs without Pencil Code. The tool allows students to change programs from blocks to text and from text to blocks, i.e., a bidirectional mode transformation. The block editor of Pencil Code is called Droplet (BAU, 2015). Droplet is based on a drag-and-drop block interface, but it also allows users to edit programs in traditional textual programming languages. Droplet was designed to load existing text programs, edit them as blocks, and save them as text again, with fidelity to the text and blocks when switching modes.

Kyfonidis, Moumoutzis and Christodoulakis (2017) created the Block C, a BBP tool for the C programming language. Block C runs on top of the C language and promotes the transition to the regular textual C programming by allowing undergraduate students to export their code and see how their block based programs are translated to textual C and how textual C is translated to block-based C. The tool helps to empower students to learn programming by guiding them to focus on the programming logic instead of the C language syntax. Results from the application of Block C in real learning scenarios reveal that novice programmers of introductory programming subjects are more productive with the use of Block C than with regular textual C programming.

Similarly, Poole (2015) designed a BBP application for creating and editing programs using the Python programming language. The purpose of the application is also to bridge the gap between programming using a simplified BBP language (e.g., Scratch or App Inventor) and a traditional textual programming language. The target audience of the application are high school or higher education students who are beginning to learn programming in Python, with either no previous programming experience or transitioning from BBP to textual programming languages. The application does not aim to support the whole Python language, but focuses on features that are commonly taught in the first phase of an introductory programming subject, allowing novice programming to focus on the fundamentals of programming rather than on the complexities of the programming language.

Moskal, Lurie and Cooper (2004) compared Computer Science students who studied Alice before or during their first programming subject to students that only took the introductory programming subject. Results show that exposure to Alice significantly improved students' grades in the subject, and their retention in Computer Science in general over a two year period. Another study in this line, conducted by Cooper *et al.* (2003) obtained similar results, showing that a CS curriculum based in Alice programming environment also resulted in improved grades and higher retention.

Still comparing BBP and text-based programming, Price and Barnes (2015) performed an experiment in which students were randomly assigned to either a text-based or a block-based environment to do small programming tasks. The experiment showed that students in the block-based environment were more focused and achieved more of the activity's goals in less time.

In order to explore the potential of BBP in the professional context, Weintrop *et al.* (2017) created a customized block-based interface with the Blockly[21] development kit for programming an one-armed industrial robot. The results showed that adults with no prior programming experience successfully programmed a virtual robot to accomplish a pick and place task. Authors state that these results indicate the potential for BBP beyond children, teenagers, students and classrooms, since it can be useful for preparing prepare workers for the increasingly technological environment of manufacturing and industrial jobs.

Despite the advantages of BBP approaches to novice programmers, professional programmers do not use languages based on blocks on their jobs. One of the reason, is that block-based manipulation has efficiency disadvantages when making small edits, as well as for creating larger and more sophisticated programs. Besides that, blocks environments have other disadvantages compared to textual programming languages, such as (BAU *et al.*, 2017): (i) blocks take more space on the screen than equivalent text code; (ii) it can be challenging to search and navigate to the relevant part of a block program, and (iii) collaboration and version control are difficult to use without a textual representation.

In order to address these issues, new BBP environments, such as Greenfoot[22], have been designed to be used by programmers to create large programs, making efficient code editing as an important design goal. Blocks could still be chosen from a palette, but a professional programmer can choose for a text-style entry of blocks. Greenfoot allows users to insert blocks by typing, use a bidirectional approach for editing code, and switch between a low-level and high-level visualization structure. The hypothesis that motivates the design of a bidirectional approach tools is that users may benefit from the learnability of blocks in one mode, while they learn syntax and get the efficiency of text in the other mode (BAU *et al.*, 2017). Moreover, it is also possible to create your own block-based programming environment with a BBP tool kit, using the libraries provided by Blockly[21], Droplet (BAU, 2015), or OpenBlocks (ROQUE, 2007).

Although programming is still not widely learned as it should be, the progress BBP has been making is fostering the conception that it is possible for everyone to learn how to code (BAU *et al.*, 2017). With the rise in popularity of BBP tools, the number of activities users can engage with BBP is growing increasingly diverse (WEINTROP; WILENSKY, 2015). At the same time, considering the advances of mobile devices, ubiquitous computing, and VR, it is also possible to: (i) create mobile applications using App Inventor (WOLBER *et al.*, 2011); (ii) create mobile applications with Augmented Reality features also using App Inventor (MOTA *et al.*, 2018); and (iii) make BBP an immersive experience with VR (VINCUR *et al.*, 2017).

Considering that mobile learning applications are capable to support both BBP and VR applications, in order to make immersive VR applications based on BBP possible for students, mobile devices can be used as hardware platforms. Today's mobile devices contain all the essential hardware components to run VR applications and, in 2018, almost 80% of the Brazilian population owns at least one mobile device[23].

The next section presents an overview of the main concepts related to mobile learning.

2.1.3 Mobile Learning

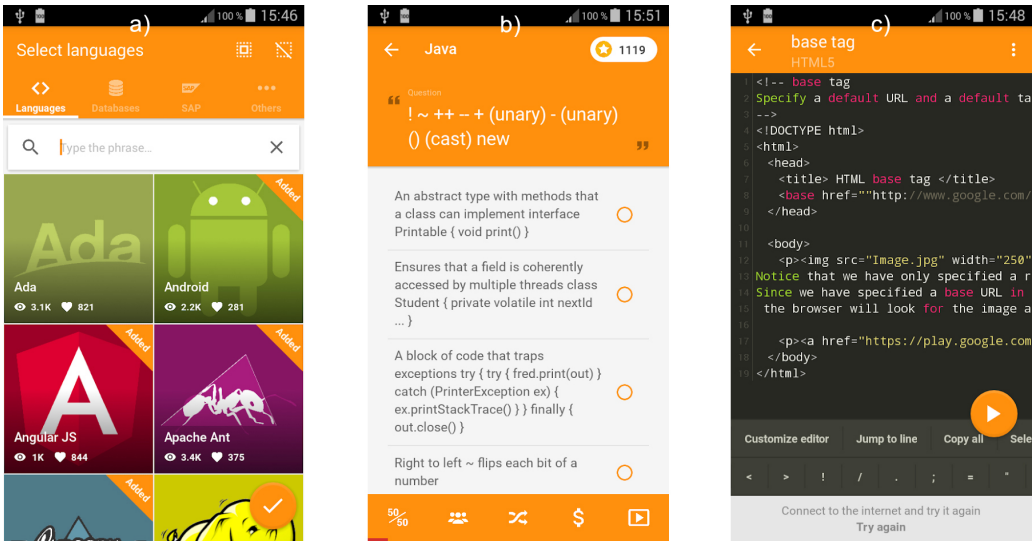
In the last years, due to the advent of ubiquitous computing and the acknowledgment of learning environments, the concept of mobile learning (m-learning) has gained distinction (WEXLER *et al.*, 2008). As a new paradigm, there are different definitions for it. According to O'Malley *et al.* (2005), mobile learning refers to any sort of learning that combines technology and mobility, i.e., any sort of learning that occurs when the student is not in a fixed place or when it is possible to take advantage of mobile devices to carry out a learning activity. As stated by Ozdamli and Cavus (2011), m-learning is an activity that allows individuals to be more productive when they consume, create or interact with information supported by mobile devices, such as smartphones, tablets, laptops, wearables, and so on.

[22] https://www.greenfoot.org
[23] https://bit.ly/39CmQsg

The variations on m-learning research made it complex to adopt a single and broad enough definition. However, the definitions need to emphasize mobility, access, immediacy, ubiquity, convenience, and flexibility. Mobile learning promotes a strong interaction among the students and with the instructors, improving convenience, flexibility, and motivation towards the process of teaching and learning (BARAN, 2014). This strong interaction characteristic is achieved due to the combination of internet, portability, and ease of integration, that provides a high degree of communication and cooperation among its users, enabling students and instructors to contribute, participate, and access learning materials at any time and any place (ECONOMIDES, 2008).

Nowadays there are several m-learning applications available for free usage in various domains, such as Languages, Mathematics, Chemistry, Music, etc. There are also different free m-learning applications available to support the teaching and learning process of programming languages, such as SoloLearn[24], ProgrammingHub[25], Mimo[26], and Learn Programming[27]. These applications support the teaching and learning of many different programming languages, such as: C++, C#, Java, PHP, HTML, CSS, Javascript, Python, R, SQL, and so on. Figure 3 presents some features of the Learn Programming application: a) the available programming languages to learn; b) theoretical multiple choice question, and c) example of an algorithm in HTML.

Figure 3 – a) available programming languages b) theoretical question c) algorithm example



Source: Extracted from Learn Programming[27]

There are also m-learning applications based on visual puzzles, games, and metaphor to develop problem-solving skills and programming skills, such as the Google Grasshopper[28], that merges BBP with a standard textual programming language to teach programming in a fun and

[24] http://bit.do/eNT2L
[25] http://bit.do/eNT2Y
[26] https://bit.ly/3dwKGqq
[27] https://goo.gl/YALKy3
[28] https://bit.ly/3rWDFUR

Figure 4 – Grasshopper textual and block-based programming interface

Source: Extracted from Grasshopper[28]

easy way. In the Grasshopper app, users move blocks of JavaScript and edit algorithms without having to worry about typing or syntax, as shown in Figure 4. They adopted an approach of using different levels with progressive challenges to teach concepts such as functions, variables, loops, and so on. Their goal is to make programming possible for everyone, even when life is busy.

Similarly, ScratchJr[29] is a mobile application based on the popular Scratch programming language that enables young children to create their own games, animations, and stories. The app is a collaboration between the DevTech Research Group at Tufts University, the Lifelong Kindergarten Group at the MIT Media Lab, and the Playful Invention Company. In addition to being for mobile devices, the ScratchJr has a redesigned interface and programming language to be appropriate for younger children's cognitive, personal, social, and emotional development, as presented in Figure 5.

In addition to those applications available for free usage, several research have been conducted on the adoption of mobile devices for the teaching and learning of programming. Jordine, Liang and Ihler (2014) proposed a mobile serious game for the teaching and learning of Java programming language, motivated by a lack of this type of application identified by the authors. The Java Tower Defense is in the prototype phase and its intent is to be an extension to the lectures and practical sessions of the traditional programming subjects, aiming to help students to improve their depth of understanding in relation to object-oriented programming (OOP). The serious game was designed to be suitable for any novice in Computer Science, with the objective to engage them in learning OOP through a mobile optimized Java keyboard and the

[29] https://bit.ly/3cSPRlo

Figure 5 – ScratchJr interface

Source: Extracted from ScratchJr[29]

adoption of traditional game design elements, fostering challenge, competition, and discovery.

Tillmann *et al.* (2012) presented the Touch Develop[30], a Microsoft novel programming environment, language, and code editor for mobile devices. According to the authors, as students are soon more likely to own a smartphone than a desktop computer or laptop, programming on smartphones can promotes an engaging learning experience, since students can do homework or practicing anytime and anywhere. With Touch Develop it is possible to write programs directly on mobile devices, using a semi-structured editor. The Touch Develop programming language adopted semantics similar to Java or C#, mixing imperative, object-oriented, and functional aspects. This similarity enables students to transfer knowledge between the Touch Develop language and traditional languages in both directions. In 2019, however, Microsoft replaced the Touch Develop platform with Microsoft MakeCode[31]. Their goal is to provide a more practical education platform, including not only the usage of mobile devices, but also physical computing devices and immersive experiences (BALL *et al.*, 2019)

In a different perspective, the catalog constructed by Marcolino and Barbosa (2016) aimed to establish quality characteristics for the development m-learning applications for the teaching and learning of programming. The authors state that the catalog is capable of addressing functional and nonfunctional requirements and connect those requirements with educational theories, with the purpose of minimize problems through the design and development of applica-

[30] https://www.touchdevelop.com/
[31] https://www.microsoft.com/makecode

tions. This catalog supported another study conducted by Marcolino and Barbosa (2017b), which deals with the project and design decisions for a software product line architecture designed only for the development of m-learning applications for the teaching and learning of programming.

Despite the benefits m-learning provides, it is a new concept in an initial stage; consequently, there still are issues to be addressed related to the development and adoption of mobile learning applications. According to Nestel *et al.* (2010) the main issues are related to: (i) processing performance; (ii) restricted battery; (iii) screen size; (iv) lack of architectural patterns, and (v) lack of pedagogical patterns. Further, due to the large number and diversity of mobile devices available in the market, the production of content for these devices became strongly dependent on the manufacturer and operational system (WEXLER *et al.*, 2008). Economides (2008) states that additionally to the technical problems, it is also necessary to consider other aspects, that are related to the social, pedagogical, and economical context.

Despite the issues associated with mobile learning, the mobile devices, that are essentially all-in-one small yet powerful computers have created a new era of potential for students. Experiences with virtual worlds, storytelling, metaphors, and learning environments in which it is possible to explore, discover, and simulate real world events, have become common for usage in desktop computers, laptops, and also mobile devices, since inbuilt sensors such as accelerometers and gyroscopes become common to these devices, in addition to their high processing power and high definition display. Mobility and the possibility of acquiring knowledge from mobile learning is one of the main benefits of using VR in mobile devices. Another benefit of mobile VR is the involvement of human senses in the interaction with the virtual environment or world (TORI; HOUNSELL, 2018).

It is important to note that, in order to present an adequate narrative to drive user's actions, stories can make users think creatively and enhance their designing skills (VINAYAKUMAR; SOMAN; MENON, 2018). Mobile devices offer unique capabilities to support such interactions through the usage of VR and storytelling elements (FOG *et al.*, 2010; COCHRANE, 2018). The evolution of hardware and ICTs have, therefore, provided the basis for the development and use of affordable mobile VR, using mobile devices as hardware (TORI; HOUNSELL, 2018).

Next, we provide an overview of the fundamental concepts related to storytelling.

2.1.4 Storytelling

According to Hamilton and Weiss (2005), storytelling is one of the oldest types of education. All around the world people have told stories as a medium of passing on culture, traditions, and knowledge. The process of constructing stories in the mind is one of the fundamental methods that humans use to make meaning. Therefore, stories permeate all aspects of learning, regardless of age. Lisenbee and Ford (2018) affirm that most of the definitions for traditional storytelling describe it as an act of telling or writing a story. A broader definition

includes different experiences that students can participate. From a story narrated by a instructor to an experience of being part of a digital story in the classroom or at home using electronic devices (LISENBEE; FORD, 2018).

Young students usually find it easier to assimilate new ideas when the ideas are presented in the form of a story, but older children and teenagers also look forward to stories to help them understand new concepts (HAMILTON; WEISS, 2005). Listening, reading, or being part of stories can be valuable learning experiences for both children and teenagers. The joyful action is motivating and it allows them to think in more sophisticated ways. Stories provide a medium in which children and teenagers can easily remember, imagine, and recreate images and ideas (WRIGHT *et al.*, 2008). Therefore, traditional and digital storytelling are a powerful literacy tool which engage students from different ages in making connections between their lives and school contents, as well as they both embrace the practicing of the 21st-century skills, including the usage of electronic devices (LISENBEE; FORD, 2018).

According to Lisenbee and Ford (2018), digital storytelling adopts electronic devices, such as an interactive whiteboard, computers, or mobile devices to provide a foundation for offering storytelling experiences in a classroom or at home. Digital storytelling definitions mostly refer to using a variety of technical tools to share narratives, images, and experiences in the form of audio, video, and so on (LISENBEE; FORD, 2018). It creates a participatory and immersive experience that allows students to enjoy learning as a dynamic and entertaining method. When students are emotionally involved with the story, it can help them develop a positive attitude toward the learning process. Even students with low motivation and weak academic skills are more likely to participate in the context of storytelling (HAMILTON; WEISS, 2005; WRIGHT *et al.*, 2008; LISENBEE; FORD, 2018).

As stated by Fog *et al.* (2010), storytelling encompasses different aspects that must be polished to the audience and situation. It is almost impossible to establish set of general rules. There are, however, guidelines that can be used. There are four elements that make up the core basis of any storytelling usage and can be used as checkpoints to develop stories (FOG *et al.*, 2010): (i) message; (ii) conflict; (iii) characters, and (iv) plot. These four elements can be mixed, matched, and applied in a variety of means depending on the context purpose of application (FOG *et al.*, 2010).

Fog *et al.* (2010) also explain that, as storytelling is about using stories to communicate messages, without a clear message, there is no reason to tell stories. The central message, or premise of the story, works as a central theme throughout the story. On the other hand, conflict is the driving force of a story. Therefore, a story needs to be set in motion by a change that disturbs its sense of harmony. The conflict forces action to be taken in order to restore harmony. It address our emotional need to bring order to chaos (FOG *et al.*, 2010).

Another important element in storytelling are the characters. Conflict marks the turning point in the story, and for the conflict to play out, it is necessary to cast interacting and compelling

characters. Once the message, conflict, and cast of characters are in place, it is necessary to define how the story progress, i.e., the plot. The flow of the story and the sequence of events are essential to provide an adequate experience to the audience. It must has a precise structure to drive the story forward and maintain audience interest (FOG *et al.*, 2010).

Among different educational usage contexts, storytelling represent a natural manner to introduce concepts of programming, such as basic concepts of objected-oriented programming (KELLEHER; PAUSCH, 2007). During the past few years, the interest among Computer Science departments, industry, and government funding sources in using mobile devices, computers, animations, simulations, and video games to draw students into the field of Computer Science has been growing. They believe that today's students are likely to be more engaged by joyful activities and interactive environments than they are by traditional Computer Science contents and teaching approaches (KELLEHER; PAUSCH, 2007). In this context, Alice introduces programming to students as a manner of creating animated movies (KELLEHER; PAUSCH, 2007). Similarly, Scratch is a graphical programming environment that allows children to create animated stories and games by assembling together graphical building blocks, each representing a different command or action (RESNICK *et al.*, 2009).

It is possible to make digital storytelling more immersive using virtual reality. The usage of storytelling with virtual reality can be classified as microworlds or virtual worlds, that designates learning environments in which it is possible to explore, discover, and simulate real world events (JONASSEN, 2007).

Next, we provide an overview of the fundamental concepts related to virtual reality.

2.2 Virtual Reality

Although the appropriate is to be more precise while defining virtual reality (VR), the term is commonly used to describe imaginary worlds that only exist in computers and our minds (JERALD, 2016). According to the Merriam-Webster[32] online dictionary, virtual[33] is defined as "being such in essence or effect though not formally recognized or admitted. Of, relating to, or being a hypothetical particle whose existence is inferred from indirect evidence" and reality[34] as "the quality or state of being real. A real event, entity, or state of affairs. The totality of real things and events". Therefore the term virtual reality is an oxymoron, a contradictory combination of words. The Merriam-Webster[32] online dictionary has also defined the full term virtual reality[35] as:

> "Virtual reality is an artificial environment which is experienced through

[32] https://www.merriam-webster.com/about-us
[33] https://www.merriam-webster.com/dictionary/virtual
[34] https://www.merriam-webster.com/dictionary/reality
[35] https://www.merriam-webster.com/dictionary/virtual%20reality

> sensory stimuli (such as sights and sounds) provided by a computer and in which one's actions partially determine what happens in the environment".

As reported by Tori and Hounsell (2018), VR definitions are focused on the technology and user perception. Jerald (2016) defines VR as a computer-generated digital environment that can be interactively experienced as a real environment, as Figure 6 illustrates.

Figure 6 – A non-immersive virtual world

Source: Vosinakis, Koutsabasis and Anastassakis (2014)

With reference to Tori, Kirner and Siscoutto (2006), VR is:

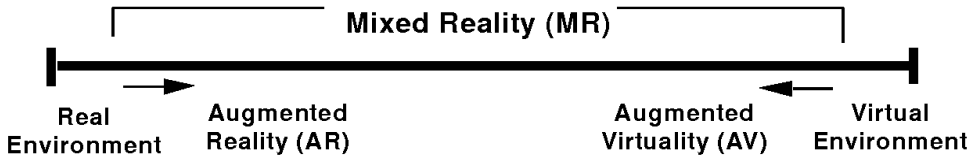
> "An advanced user interface to view, navigate, and interact in three-dimensional environments in real time. In addition to the visualization itself, the user experience can be enriched by the stimulation of the other senses, as touch and hearing".

The consolidation of the term of VR only occurred in the 1980s, based on research conducted by Jaron Lanier, a computer scientist, visual artist, and composer who associated these two opposing concepts into a new one. The VR term is capable of precisely represent the essence of this technology: the alliance of the real with the virtual, but not in a sense of mixing them (RODELLO *et al.*, 2010; TORI; HOUNSELL, 2018). However, it was long before the consolidation of the term that appeared the first research and the first results that created the basis for virtual reality. Sutherland (1965), the creator of one of the world's first VR system in the 1960s, stated:

> "The ultimate display would, of course, be a room within which the computer can control the existence of matter. A chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal".

Taking this into account, an ideal VR environment enables users to navigate around objects and touch them as if they were real, although this is still somewhat complex to create (JERALD, 2016). Real and virtual elements are separately handled, since the objective of VR is to remove user's perception of the real world and let available only the perception of the virtual environment. In the 1990s, the concept of augmented reality (AR) emerged and the combination of real and virtual became a possibility (TORI; HOUNSELL, 2018). The study conducted by Milgram *et al.* (1994) presented the concept of the reality-virtuality continuum, as Figure 7 reveals.

Figure 7 – Reality-virtuality continuum



Source: Milgram *et al.* (1994)

The case at the left of the continuum defines any environment consisting exclusively of real objects, including what might be observed when seeing the real world scene either directly in person or through some sort of digital display. The case at the right defines environments consisting exclusively of virtual objects, such as computer graphic simulations, virtual environments, or virtual worlds, that may or may not represent an existing environment, either based on a display or using immersive equipment. It is also possible to define a generic mixed reality (MR) environment, in which real world and virtual objects are presented together, at the same time and can be seen through the same equipment or display. Augmented reality is obtained when users feel present in the real world and can interact with virtual elements properly registered, i.e., positioned in a coherent way on the real physical space (MILGRAM *et al.*, 1994). The fundamental concepts of AR are presented and compared to VR in the Section 2.3.

VR is an alternative reality created by computer, but it is perceived by human's sensory systems in the same way as the real world. Consequently, VR is able to emotionalize, teach, entertain, and respond to external actions without the need to exist in a tangible way. Besides that, tangibleness has already been part of a few sophisticated virtual worlds and environments, enabling those applications to be further indistinguishable from reality (TORI; HOUNSELL, 2018).

Nowadays, a large part of this can be currently done at a low cost, being only necessary to have a desktop compuer, laptop, or a mobile device together with a head-mounted display (HMD) or VR adapter to be able to interact with virtual objects in non-immersive or immersive environments with high-definition graphics. The area of Education has much to gain using VR applications, as well as the area of Science, since VR is capable to create to users a new way of visualizing concepts and experiencing situations. VR is capable of efficiently present: (i) abstract concepts, such as programming concepts; (ii) behavior of elements that can not be seen, such as a computer memory; (iii) behavior of very large or very small elements, such as galaxies or atomic structures; and (iv) other educational and Science concepts (TORI; HOUNSELL, 2018).

As briefly mentioned, the concept of immersion is highly related to VR applications. Also, as important as the concept of immersion, is the concept of presence. These both concepts are presented next.

2.2.1 Immersion and Presence

The concepts of immersion and presence are closely related to VR and also to each other. Immersion is objective, while presence is subjective. It is possible to measure and compare the immersion quality of VR applications, but even with the most immersive environment it is not possible to guarantee that users will actually feel present while using it (TORI; HOUNSELL, 2018).

Immersion refers to how accurate VR is for providing to the user an illusion of reality, i.e., it is the level a VR world or environment is able to stimulates the user's sensory receptors (SLATER; WILBUR, 1997). According to Tori (2010a), immersion parameters are mainly focused on the sense of sight, the most important for VR, but immersion can also be enhanced with other senses such as hearing and touch. Jerald (2016) characterized a set of six variables that define the level of immersion:

- **Extensiveness** is the range of sensory modalities presented to the user (e.g., visuals, audio, and physical force).
- **Matching** is the congruence between sensory modalities (e.g., appropriate visual presentation corresponding to head motion and a virtual representation of one's own body).
- **Surroundness** is the extent to which cues are panoramic (e.g., wide field of view, spatialized audio, 360 degrees tracking).
- **Vividness** is the quality of energy simulated (e.g., resolution, lighting, frame rate, audio bitrate).
- **Interactability** is the capability for the user to make changes to the virtual environment or world, the response of virtual entities to the user's actions, and the user's ability to influence future events.

- **Plot** is the story, the consistent portrayal of a message or experience, the dynamic unfolding sequence of events, and the behavior of the virtual environment or world and its entities.

Immersion is only part of the VR experience as it takes a human to perceive and interpret the presented stimuli, it can lead the mind but cannot control the mind. How users subjectively experience the immersion is known as presence (JERALD, 2016). According to Slater and Wilbur (1997), presence is a state of consciousness, it is related to the psychological perception that the user is within an virtual environment. On the other hand, Lombard and Ditton (1997) define presence as "the perceptual illusion of non-mediation". As specified by Jerald (2016), there are four types of illusion of presence: (i) the illusion of being in a stable spatial place; (ii) the illusion of self-embodiment; (iii) the illusion of physical interaction; and (iv) the illusion of social communication.

(SLATER, 2018) complements that presence is not about belief. Presence is the illusion of being there, notwithstanding that knowing for sure that they are not. It is a perceptual but not a cognitive illusion. This is the real power of VR, and, like any illusion, even though one knows it is an illusion, this does not change ones perception or response to it.

As presence is a subjective perception, it is difficult to make an objective assessment of how present a user is feeling in an environment. For this reason, the most widespread technique of measuring presence perception is through questionnaires. There are standardized questionnaires in this field to measure both immersion and presence (FU; SU; YU, 2009; LAARNI *et al.*, 2015). The usage of prototypes and standardized techniques to evaluate presence make it possible to evaluate the degree of impact of project decisions on the presence perception (TORI; HOUNSELL, 2018).

Immersion and presence are important aspects to the VR applications. This is due to the enhancements in interaction and engagement provided by experiencing activities in a practical way. All sensory stimuli are important for feeling presence in the plot of a virtual environment. Devices such as HMDs and projection rooms (digital cave) are used to produce the feeling of presence. Immersion and presence are also related to the ability of the VR application to detect user inputs and instantly modify the virtual environment and the actions on it (TORI; HOUNSELL, 2018).

An application that supports users to experience virtual environments using the whole body, enabling them to bend down to look underneath something, reaching out and looking around an object, would be at a higher level of immersion than one that just afforded looking at a display, since as soon as one turns their head away from the display they are no longer perceiving the virtual world (SLATER, 2018).

As stated by Rodello *et al.* (2010), immersion and presence are achieved mainly through the usage of special equipment that enhance the sense of sight, hearing, and touch. It is also possible to obtain different levels of immersion and presence with conventional devices, however,

in mostly cases, it will neither be possible to acquire full immersion nor presence. As a result, VR applications classified as immersive, typically adopt equipment such as HMDs and tracking and motion capture devices. VR applications classified as non-immersive are usually directly experienced in a digital display, which interaction through conventional devices will possibly limit actions within the virtual environment. The special equipment referred is better presented and discussed in Subsection 2.3.2.

Besides the concepts of immersion and presence, it is also essential to understand how VR applications work, as well as their technical and development aspects. These contents are approached next.

2.2.2 Virtual Reality Applications

As explained by Tori and Hounsell (2018), VR applications have two basic components: hardware and software. Hardware includes input devices, displays, processors, and networks. Software includes software development tools, virtual objects database, interaction functions, and the input and output interface.

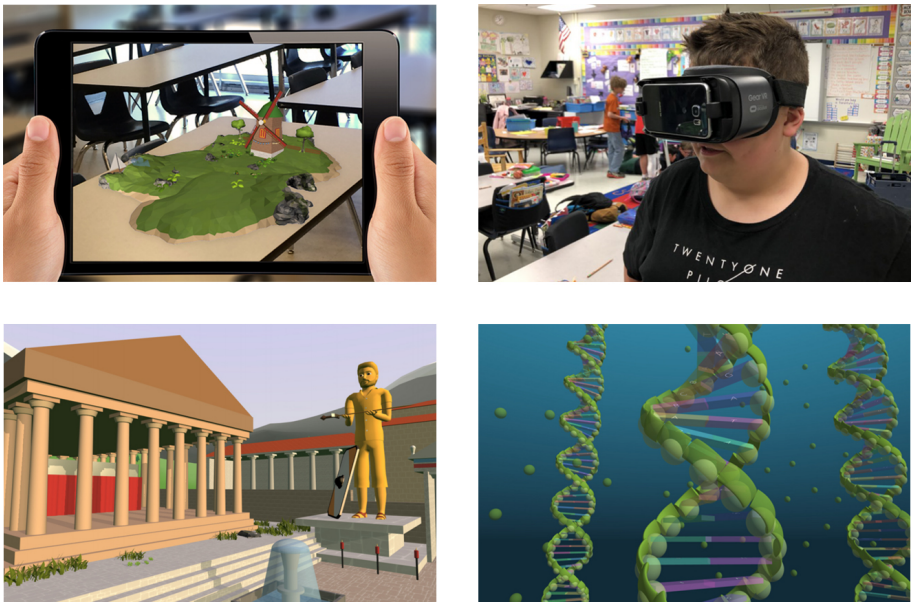
VR applications should have input and output interfaces to interact with users. The input interface is mostly used to track the position and orientation of the user's head and hands. The output interface is used for the visualization, emission of sound, and touch reaction. VR applications are complex and involve real-time interactions between many hardware and software components. The software development tools may involve: (i) programming languagues, as C++, C#, Java, or Python; (ii) graphics library, as OpenGL, WebGL, or X3D; and (iii) game engines, as Unreal Engine, CryEngine, or Unity 3D (TORI; HOUNSELL, 2018).

Researchers have explored the benefits and applications of VR in different scenarios. VR possesses potential and its application in education has seen many research interest lately. All levels of education need to offer students activities that involve, thrill, arouse curiosity, transport to other realities and dimensions, simulate, exhibit abstract concepts, and are playful (TORI; HOUNSELL, 2018).

As an example of virtual environment that combines virtual and augmented reality, the CoSpaces Edu[36], is an educational web platform that enables students and instructors to build their own VR and AR educational projects. The goal of the CoSpaces Edu is to empower students and instructors of primary and secondary school to become creators and prepare them for the future of the education. From the browser, using a BBP language, CoBlocks, JavaScript, or TypeScript, CoSpaces Edu allows users to create, explore, and experience their creations using technology to connect with their curriculum. As Figure 8 shows, the educational technology provides numerous practical applications to STEM subjects, social sciences, language and literature, and art.

[36] https://cospaces.io/edu/

Figure 8 – CoSpaces Edu



Source: Extracted from CoSpaces Edu[36]

VR and AR are supported on different hardware and software platforms and can be built using several software development tools, but the game engines are the most viable option nowadays, bearing in mind the ease of use of their development environments and support for most operation systems, mobile devices, and HMDs on the market (TORI; HOUNSELL, 2018; JERALD, 2016). The development of virtual environments involves 3D modeling, coding interactions, preparation and manipulation of textures, manipulation of sound, elaboration of animations, and so forth. As a run-time support, the VR software should: (i) interact with special equipment; (ii) take care of user interface; (iii) deal with visualization and user interaction; (iv) control the simulations and/or animations of the virtual environment, and (v) implement network communication for remote collaborative applications (TORI; HOUNSELL, 2018).

Despite the advancement of hardware and software, tools, and development platforms, there are still challenges associated with the construction and use of VR applications, such as: (i) lack of concrete design guidelines and examples; (ii) difficult to design for the physical aspect of immersive experiences; (iii) difficult to design story-driven immersive experiences, and (iv) user testing and evaluation challenges (ASHTARI *et al.*, 2020).

Despite VR and AR are both alternative realities and can benefit from similar hardware and software, they present substantial differences. Next, we briefly present the fundamental concepts related to AR.

2.3 Augmented Reality

Through the years, augmented reality has been defined in several manners. In accordance with Milgram *et al.* (1994), AR is the combination of the real and virtual worlds at some point in the spectrum that connects completely real environments to completely virtual environments. Insley (2003) affirms that AR is a real world improvement with texts, images and virtual objects generated by computer. Rodello *et al.* (2010) reported that AR allows the user to view and interact with virtual objects overlapping or composing a scene with the real world, hence, real and virtual objects will coexist in the same space.

The definition proposed by Azuma *et al.* (2001) is the most detailed, referring to the components of AR applications as well as to AR features:

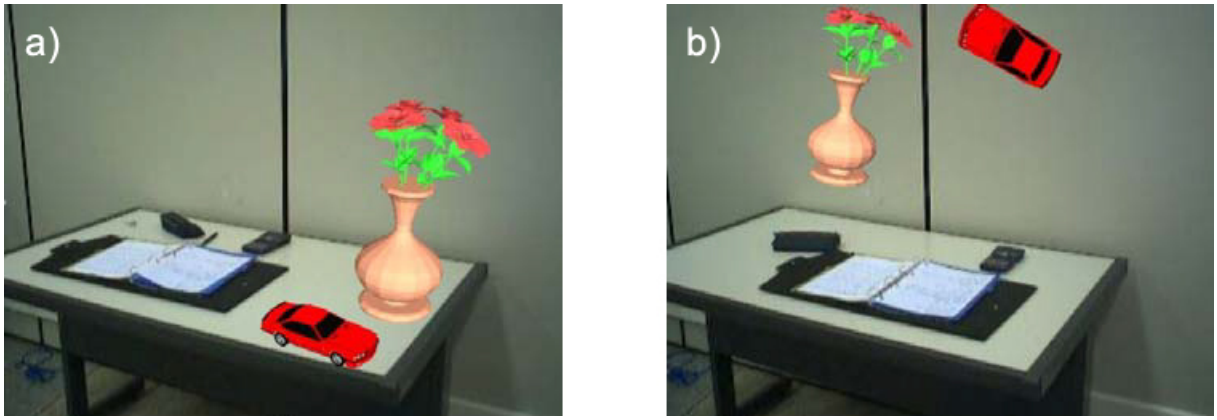
> Augmented reality is a system that supplements the real world with virtual objects generated by computer that coexist in the same space and present the following properties: combines real and virtual objects in the real environment; runs interactively in real time; aligns real and virtual objects with each other; and applies to all senses, including hearing, touch, and smell.

As explained by Tori and Hounsell (2018), besides researching on VR, Sutherland (1965) was also researching on AR, therefore, it is attributed to the same researcher the creation in 1968 of the first device prototype that allowed to merge computer-generated 3D images with real images. Although the first scientific article that used the term augmented reality was wrote by Caudell and Mizell (1992), being credited to them the creation of the term AR. The usage of the concept of AR around 1968 and its actual appearance in 1992 were the two milestones of AR's creation.

The AR technology enhances the physical environment with computationally synthesized objects, allowing the coexistence of real and virtual objects (KIRNER; SISCOUTTO, 2007). As specified by Tori and Hounsell (2018), AR was considered as a type of VR, although they have been indistinctly developed. AR is currently considered as a technology related to VR, and not a type of VR. Differently from VR, which transports the user to a whole new virtual environment, making the physical and local environment completely abstracted, AR maintains the real world references, only including virtual elements to the user space, as Figure 9 illustrates.

The AR goal is to enable users to interact with the real world and virtual elements in a natural and intuitive way, without the need for training or adaptation. This interaction can be either directly (e.g., with the user's hand or body) or indirectly (e.g., aided by some interaction device) (TORI; HOUNSELL, 2018). The utopia of the AR scenario is the creation of an environment where the user can not distinguish the real world from the virtually augmented objects (RODELLO *et al.*, 2010).

Figure 9 – a) AR with virtual vase and toy car coherent to the table b) AR with unregistered or inconsistent virtual objects



Source: Kirner and Siscoutto (2007)

Bimber and Raskar (2006) emphasize that: (i) AR enriches the real world with virtual objects, while VR is fully computer-generated; (ii) in the AR environment, users maintain the sense of presence in the real world, while in VR, the visual sensation is controlled by the application, and (iii) AR needs a mechanism to combine real and virtual, while VR needs a mechanism to integrate the user within the virtual world.

In order to maintain the coherence of the physical world references for users, AR applications must be able to identify where virtual elements should be placed and how they should be presented. This process is named tracking, it is essential for AR, and can be achieved through different techniques (TORI; HOUNSELL, 2018). With respect to AR visualization methods, the two main techniques are optical see-through and video see-through (MILGRAM *et al.*, 1994; RODELLO *et al.*, 2010).

The classification of AR applications in relation to the tracking and visualization techniques is presented next.

2.3.1 Tracking and Visualization

As mentioned, AR applications can be classified by their tracking technique. This tracking can be based on computer vision or sensors. In this regard, AR applications based on computer vision require to capture and process the real world content to include and track virtual resources. In contrast, AR applications based on sensors create the association of virtual resources with the real world based on information from sensors that read the environment and guides the composition of real and virtual (RODELLO *et al.*, 2010).

As stated by Tori and Hounsell (2018), although there are still problems and inaccuracies with AR based on vision, it is still the most widely used technique. Its tracking methods are robust, flexible, and easy to use. On the other side, AR based on sensor brings more accuracy,

less latency and jitter, and is better to deal with limitations (e.g., dirt, poor illumination, lighting variation, scenes with many objects and occlusion, and so on).

Another way of classifying AR applications is related to the visualization approach, which can be optical see-through, i.e., the virtual object is projected on the observation of the real, or video see-through, i.e., the virtual object is inserted in the reproduction of the real (TORI; HOUNSELL, 2018).

Video see-through AR represent a window on the world (WoW). In this category, the application's contents are visualized through conventional digital displays and screens of desktops, laptops, and mobile devices. The augmented video stream visualized by users was previously captured by a camera and processed to include virtual resources. While simplicity is one of the advantages of video see-through applications, the resolution of the camera and device's display is the weakness of this type of AR applications. Another weakness is the visual fatigue caused by viewing an artificially illuminated display for long periods (RODELLO *et al.*, 2010).

Optical see-through AR is characterized by the ability to see through a display directly to the real world. The virtual objects are projected "directly" in the user's eyes by using translucent lens to combine computer generated graphics onto directly viewed real world scenes. Such displays are already a mature technology in some areas, mostly in military aviation systems and in AR glasses (MILGRAM *et al.*, 1994). Its quality tends to be superior in comparison with video see-through applications, but due to the freedom provided the user, the main weakness of optical see-through applications is the difficulty in tracking the environment, that can causes difficulties in the composition of the scene. The cost of this type of applications and equipment could be as well a major obstacle (RODELLO *et al.*, 2010).

Next, we present an overview of VR and AR common hardware and special equipment.

2.3.2 Virtual and Augmented Reality Hardware

Over the years, VR and AR applications have been supported by the evolution of computers and mobile devices hardware, enabling the creation and use of applications that only existed in academic, military, and industrial environments. The essential hardware for running VR and AR applications are those currently present in the modern mobile devices, desktop computers, and laptops, together with their peripheral devices and components, such as displays, touchscreens, mouses, keyboards, cameras, inertial sensors, microphones, speakers, and so forth (TORI; HOUNSELL, 2018).

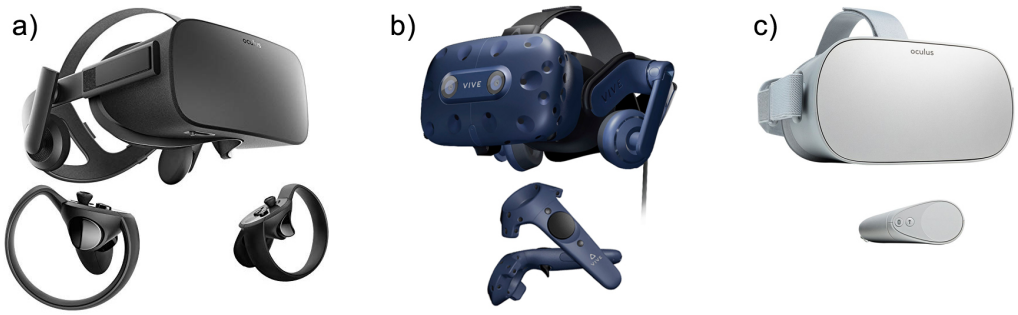
However, as explained by Tori and Hounsell (2018), these common peripheral devices are appropriate only for basic virtual environments. Since there is a increasingly growing in the development and availability of accessible equipment that allows users to act more naturally when using VR and AR applications, i.e., similarly to what occurs in the real environments, for each type of experience and situation, specific devices can be adopted. The special equipment for VR

and AR applications include a variety of devices that help users to view and communicate with the application, such as: (i) positional and rotational tracking; (ii) head-mounted displays (HMD) or VR adapters; (iii) joysticks; (iv) 3D mouses; (v) electronic gloves; (vi) speech recognizer, and so on. AR applications are capable of using typical VR equipment, however, the primary difference is that a camera could be one of the fundamental components only for AR (JERALD, 2016; TORI; HOUNSELL, 2018).

Considering that VR is about psychologically being in a place different than where one is physically located, where that place may be a replica of the real world or may be an imaginary world that does not exist and never could exist, the principle of HMDs is to project images of the virtual environment directly into the user's eyes through two small monitors using the principle of stereoscopy (JERALD, 2016; RODELLO *et al.*, 2010). The stereoscopy (or stereoscopics) includes depth to the projection screens of virtual environments and consequently makes them more realistic as the way we see the real world. It requires support of the VR application and lenses to be able to present a different image for each eye. In practice, the brain is deceived from two different viewpoints artificially created, one for each eye, as well as human 3D vision (TORI; KIRNER; SISCOUTTO, 2006; KIRNER; SISCOUTTO, 2007).

The usage of HMDs promotes freedom for users to move around in the scene, having their movements tracked by different types of tracking mechanisms. HMDs place the user fully immersed in the virtual world since besides the stereoscopy, there is no possibility to see parts of the real world through the equipment (TORI; HOUNSELL, 2018). Figure 10 illustrates three examples of sophisticated HMDs.

Figure 10 – a) Oculus Rift b) HTC Vive PRO c) Oculus Go



Source: Oculus (2019b), Vive (2019), Oculus (2019a)

According to Powell *et al.* (2016), advancements in the power of graphics processing of mobile devices have made them capable to display virtual worlds and environments in real time. In order to obtain stereoscopic vision and allow immersion, it is necessary to use an HMD. In this case, the Google Cardboard can be used, which is presented in Figure 11. This HMD was launched in 2014, is made of cardboard, uses special lenses for stereoscopy, and the existing sensors in the mobile device itself to reproduce the virtual environment. Even with several limitations, as reduced field of view, graphics fidelity dependent on the mobile devices, and

limitation of interaction with the virtual environment, it is still one of the most accessible HMDs on the market, since it can be used with any modern smartphone (TORI; HOUNSELL, 2018).

Figure 11 – Google Cardboard

Source: Google (2020)

HMDs represent the user's eyes within the virtual environment. In addition, all movements of the user's head can be transferred to the virtual environment, enabling the users to acting according to six degrees of freedom (6 DoF), i.e., rotation and translation in X, Y, and Z Cartesian axes (RODELLO *et al.*, 2010). The use of head-mounted displays is not exclusive to VR applications, it is also possible to use this type of viewer for AR applications. The HMD will essentially needs an attached camera, to be capable of showing the augmented real world with the virtual objects properly positioned (video see-through) (TORI; HOUNSELL, 2018).

Considering the presentation of the fundamental concepts of virtual reality and augmented reality, as well as the main concepts related to the teaching and learning of programming, the next chapter presents and discusses a mapping study aimed to relate these two fields.

2.4 Virtual and Augmented Reality in the Teaching and Learning of Programming: A Systematic Mapping Study

In this Section, we present a systematic mapping study (SMS) of the literature conducted for the characterization of the current scenario of VR, AR, and/or MR applications in the teaching and learning of programming. The objective is to understand the research area as well as use the obtained results to find out whether there is a lack of studies.

Contents of this Section were previously presented in a paper entitled *"Virtual and Augmented Reality in the Teaching and Learning of Programming: A Systematic Mapping Study*,

published in the proceedings of the XXX Brazilian Symposium on Computers in Education (SBIE 2019) (AVELLAR; BARBOSA, 2019).

We followed the guidelines proposed by Kitchenham and Charters (2007) and Petersen, Vakkalanka and Kuzniarz (2015) to conduct this mapping study. The method includes three main phases, namely: (i) planning the review; (ii) conducting the review, and (iii) reporting the review.

2.4.1 Planning

The planning phase of a mapping study requires the definition the research objectives and research questions, as well as the description of the search strategy and criteria for selection of studies.

Research Objectives

This mapping study main objective is to identify, classify, and analyze studies that adopt VR and/or AR for supporting the teaching and learning of programming. We also aim to collect evidence of gaps for conducting new research.

Research Questions

Aiming to achieve the research objectives, we defined the following research questions (RQ):

RQ 1: What is the Virtual, Augmented, and/or Mixed Reality setup?

- **RQ 1.1:** What is the hardware platform?
- **RQ 1.2:** What are the specifications related to VR/AR/MR?
- **RQ 1.3:** How is the VR/AR/MR content visually presented to the user?

RQ 2: What is the target audience?

RQ 3: What topics of programming are covered?

RQ 4: What software development tools are used?

The method proposed by Petticrew and Roberts (2006), entitled PICOC (population, intervention, comparison, outcome, context), was used for helping the recognition of relevant information for the mapping study:

Population: Primary studies related to teaching and learning of programming using VR, AR, and/or MR.

Intervention: Identify how VR/AR/MR are being used in educational applications for the teaching and learning of programming, together with which public use these applications, what programming topics are addressed, and how the applications are developed.

Comparison: Not applicable.

Outcome: An analysis of the primary studies that adopt VR/AR/MR for the teaching and learning of programming and the trends and gaps to conduct new research.

Context: Academia and industry.

Search Strategy

We performed the search for studies as a two-step process: an automatic search in databases and a manual search specifically for Brazilian's studies.

The automatic search was performed in four different digital libraries and electronic search engines recommended for research in Computer Science and Education, shown in Chart 1. We selected these databases since they are among the more frequently used in systematic reviews (ZHANG; BABAR; TELL, 2011).

Chart 1 – Search databases

Source	Web address
ACM Digital Library	https://dl.acm.org
Engineering Village	https://www.engineeringvillage.com
IEEE Digital Library	https://ieeexplore.ieee.org
Scopus	https://www.scopus.com

In order to perform the automatic search, we defined the search string including the main keywords from the research objectives, as presented in Figure 12. The search string was used against primary studies titles, keywords, and abstracts.

Figure 12 – Search string

("teaching" OR "learning")
AND
("programming")
AND
("virtual reality" OR "augmented reality" OR "mixed reality")

After, we performed a manual search for relevant studies in Brazilian's conferences and journals (RBIE, RENOTE, CBIE, SVR, SBGames, CTRL+E). The time window considered was chosen accordingly to time window of the studies retrieved by the automatic search (2008–2018). We also performed a search with the same parameters on Google Scholar.

Selection Criteria

As Chart 2 shows, the following selection criteria were adopted for the screening of each obtained study. The selection criteria helps the inclusion of studies that would potentially answer our research questions and the exclusion of non-relevant, incomplete, unavailable, or duplicated studies.

Chart 2 – Inclusion and exclusion criteria

Type	Criterion
Inclusion	**I1:** Primary studies that address virtual, augmented and/or mixed reality for the teaching and/or learning programming.
Exclusion	**E1:** Primary studies that do not involve the research questions.
	E2: Primary studies that are neither in English, nor in Portuguese.
	E3: Primary studies that are not available for download.
	E4: Duplicated primary studies.
	E5: Primary studies presented as a table of contents, short course description, or summary of a conference or workshop.

2.4.2 Conducting the mapping

Conducting the mapping study consists of setting the plan into practice, including the retrieval and selection of primary studies and the data extraction. The search was performed in November 2018. Figure 13 presents an overview of the conduction phase and the number of primary studies at each phase.

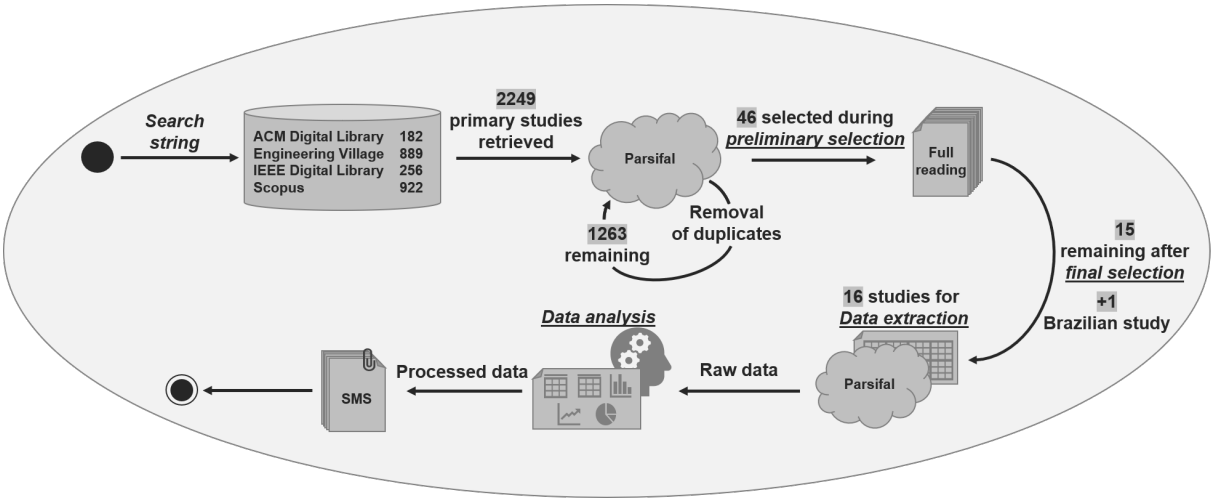
Primary Studies Retrieval and Selection

We obtained 2249 studies from the automatic search. After removing duplicates, 1263 studies remained. Figure 13 provides an overview of the conduction phase of the mapping study.

Throughout the preliminary selection, we read title, keywords, and abstract of each one of the 1263 primary studies. A total of 46 potentially relevant primary studies were selected during this selection. After a full reading of those 46 studies, we were left with 15 studies. After the manual search for Brazilian primary studies, we selected one study accordingly to our selection criteria, therefore this mapping study obtained 16 primary studies to analyze. We used Parsifal[37] to support the management of the retrieved studies. Parsifal[37] is an online tool designed to support researchers to perform systematic literature reviews (SLR) and systematic mapping studies within the context of Computer Science.

[37] https://parsif.al/

Figure 13 – Systematic mapping phases and results



Source: Author

Data Extraction

Data extraction was also made with the support of Parsifal[37]. We extracted the following data from the selected primary studies: title, authors, bibliographic reference, hardware platform of the VR/AR/MR application, operation of the application, specifications of VR/AR/MR usage, target audience, concepts of programming approached, and software development tools used to create the application.

2.4.3 Analysis of Results

In this phase, the data extracted from the primary studies is processed towards answering the research questions. The goal is to understand the patterns and relations between the selected primary studies, as well as classify and categorize them in a way that is possible to draw conclusions.

Chart 3 summarizes the selected studies, presenting the study title, reference, and ID that will be used along this Chapter for identification.

Chart 3 – Primary studies selected

ID	Reference	Title
S1	Ortega *et al.* (2017)	iProg: Development of Immersive Systems for the Learning of Programming
S2	Vosinakis, Koutsabasis and Anastassakis (2014)	A platform for teaching logic programming using virtual worlds
S3	Chandramouli, Zahraee and Winer (2014)	A fun-learning approach to programming: An adaptive Virtual Reality platform to teach programming to engineering students
S4	Oh *et al.* (2013)	The Digital Dream Lab: Tabletop puzzle blocks for exploring programmatic concepts
S5	Vincur *et al.* (2017)	Cubely: Virtual reality block-based programming environment
S6	Teng and Chen (2012)	An augmented reality environment for learning openGL programming
S7	Mota *et al.* (2018)	Augmented reality mobile app development for all
S8	Stigall and Sharma (2017)	Virtual reality instructional modules for introductory programming courses
S9	Sharma and Ossuetta (2017)	Virtual reality instructional modules in education based on gaming metaphor
S10	Masso and Grace (2011)	Shapemaker: A game-based introduction to programming
S11	Díaz, Albarrán and Calero (2008)	Role-play virtual environments: Recreational learning of software design
S12	Chandramouli and Heffron (2015)	A Desktop VR-based HCI framework for programming instruction
S13	Figueiredo, Chacón and Gonçalves (2016)	Learning programming and electronics with augmented reality
S14	Mesía, Sanz and Gorga (2016)	Augmented reality for programming teaching: Student satisfaction analysis
S15	Magnenat *et al.* (2015)	Enhancing Robot Programming with Visual Feedback and Augmented Reality
S16	Carvalho, Aguiar and Dantas (2017)	*Ensino da estrutura de repetição For em Python com realidade aumentada através do Aurasma*

Figure 14 reveals the temporal distribution of the studies along with their country.

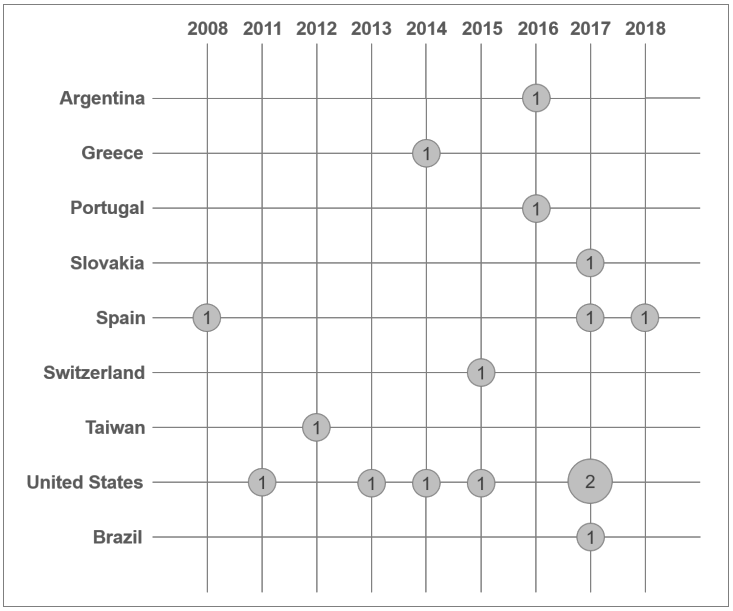
RQ 1: What is the Virtual, Augmented, and/or Mixed Reality setup?

We considered setup as the application's main attributes, such as: (i) application's hardware platform; (ii) VR, AR, and/or MR technical specifications; and (iii) how the VR, AR, and/or MR content is visually presented to users.

RQ 1.1: What is the hardware platform?

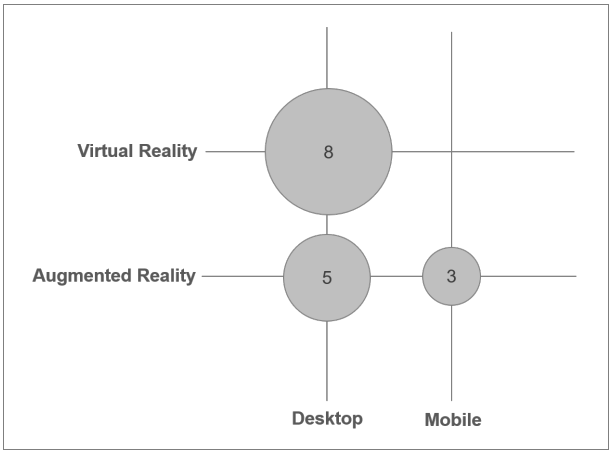
Besides the even distribution between studies that implement VR or AR, Figure 15 also shows that desktop is the most common hardware platform for both VR (S2, S3, S4, S5, S8, S9, S11, S12) and AR (S1, S6, S7, S10, S14) applications. We could not find primary studies that implement any other level of Mixed Reality, considering the reality-virtuality continuum (MILGRAM *et al.*, 1994).

Figure 14 – Selected studies distribution per year and country



Source: Author

Figure 15 – VR and AR distribution per platforms



Source: Author

Studies that implement AR contemplate the two most common hardware platforms available nowadays for VR and/or AR applications, i.e., desktop computers and mobile devices. Even though the increasing popularization, low cost, and flexibility of development for mobile learning applications with VR and/or AR, we could not find any mobile VR application. On the other hand, we obtained three mobile AR applications (S13, S15, S16).

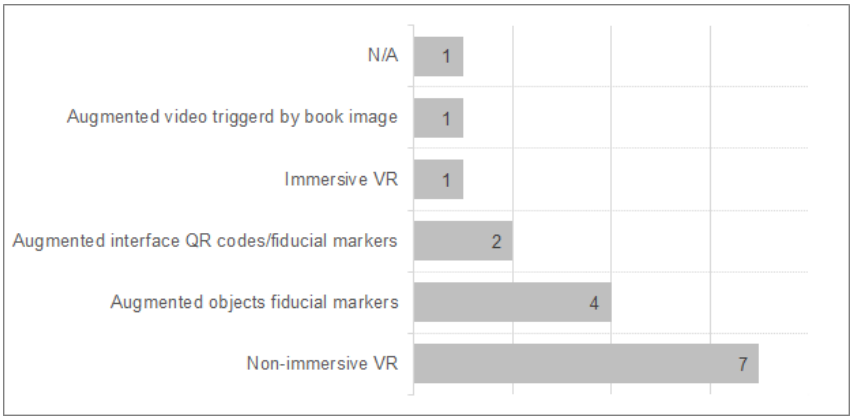
RQ 1.2: What are the specifications related to VR/AR/MR?

According to the VR definition, it is possible to characterize these applications regarding to the level of immersion and presence provided. In contrast, AR applications can be classified

according to the tracking mechanism used and how the user visualizes the application. In this regard, six categories were created to classify the studies, as shown in Figure 16.

The most common specification is non-immersive VR (S2, S3, S4, S8, S9, S11, S12). The second most common category is video see-through AR based on markers (S6, S7, S14, S15, S16). Besides the two leading specifications mentioned, one study employed an augmented interface through QR codes (S10), one study used augmented video triggered using images from a book (S13), one study approached the use of immersive VR using the HMD HTC Vive (S5), and lastly, although mentioning the use of AR to view the program commands stack while collaboratively programming, this single primary study does not provide any information concerning AR specifications (S1).

Figure 16 – AR and VR applications specifications distribution



Source: Author

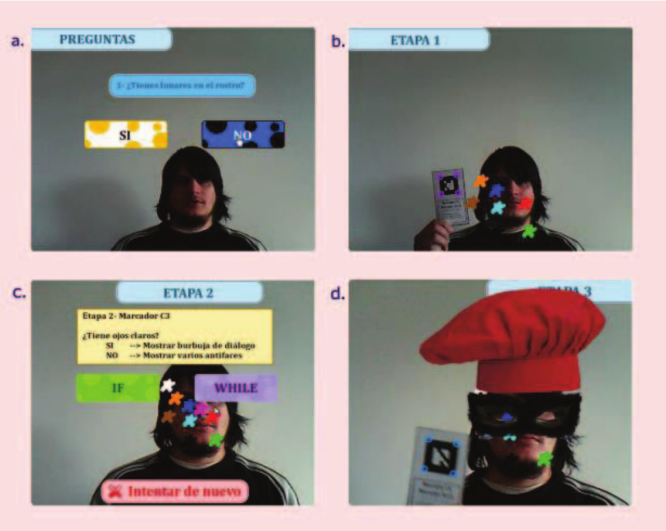
RQ 1.3: How is the VR/AR/MR content visually presented to the user?

AR applications were supported by teaching and learning methods based on the visualization and manipulation of augmented objects, which can represent abstract programming concepts (S1, S6, S7, S10, S14, S16) (Figure 17), provide real-time feedback (S15), and show augmented videos (S13). VR applications were presented as virtual environments for straightforward objects manipulation, such as value entries, changes in objects and texts format, positioning, size, and color (S3, S4, S12) (Figure 18); and as virtual worlds, allowing more interactive experiences, fostering exploration and guiding users using different narratives (S2, S8, S9, S11). The only immersive virtual world obtained is focused on solving programming puzzles in a three-dimensional game-like sandbox (S5).

RQ 2: What is the target audience?

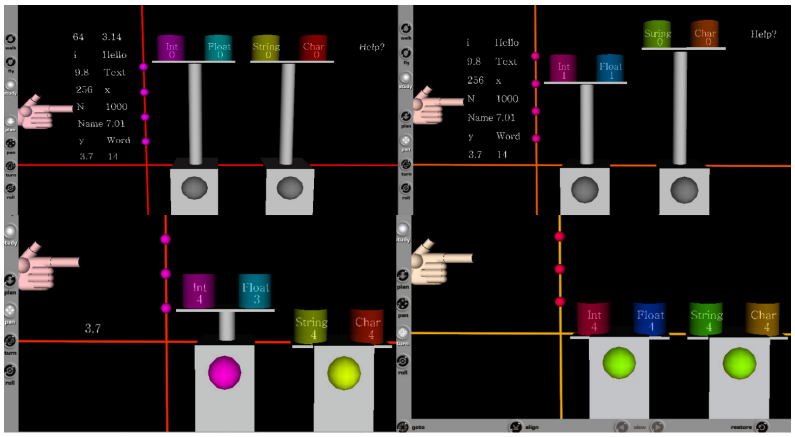
A wide range of target audiences was found for AR and VR applications that help in the teaching and learning of programming, as shown in Figure 19. We classified the target audience

Figure 17 – Augmented reality for programming teaching (S14)



Source: Mesía, Sanz and Gorga (2016)

Figure 18 – A Desktop VR-based HCI framework for programming instruction (S12)



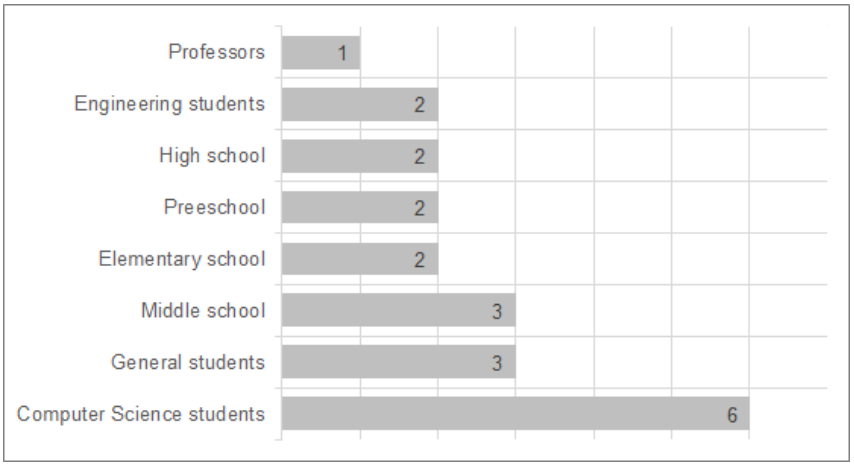
Source: Chandramouli and Heffron (2015)

with respect to education stage and age. It goes from preschool children to undergraduate and graduate adults.

Despite there are flexibility among audiences, the primary studies are frequently associated to singular groups, such as younger students from preschool (S4, S15), elementary school (S5, S15), middle school (S5, S13, S15), high school (S5, S15), and undergraduate and graduate adults (S2, S3, S6, S8, S9, S12, S14, S16). The audience also included professors (S7) and, due to the lack of accurate information from some primary studies in regard to the target audience, we also created the *General Students* category (S1, S10, S11).

Although children and teenagers have obtained significant attention from the researchers, most applications aim to support Computer Science students, especially those enrolled in introductory programming courses.

Figure 19 – Target audiences distribution

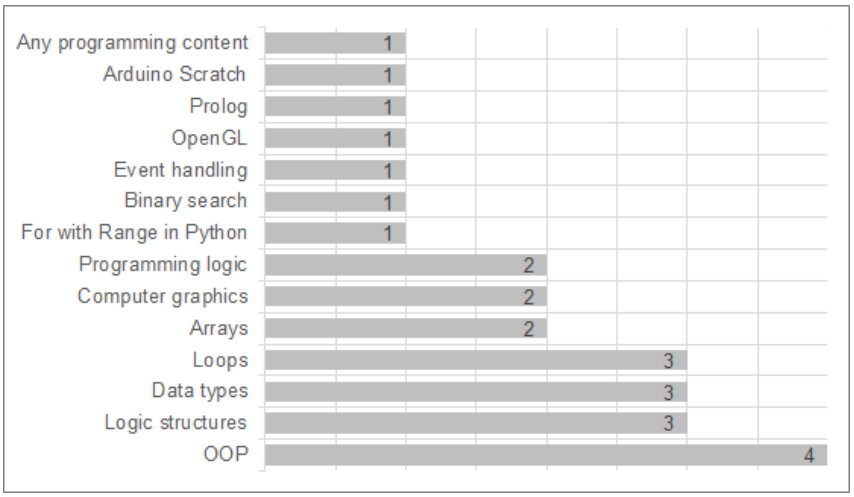


Source: Author

RQ 3: What topics of programming are covered?

Despite we obtained studies that approach a variety of programming topics, most of them demonstrated a common interest towards the fundamental concepts of programming. The majority of studies are focused on the teaching and learning of more than one content related to programming (Figure 20).

Figure 20 – Selected studies programming content distribution



Source: Author

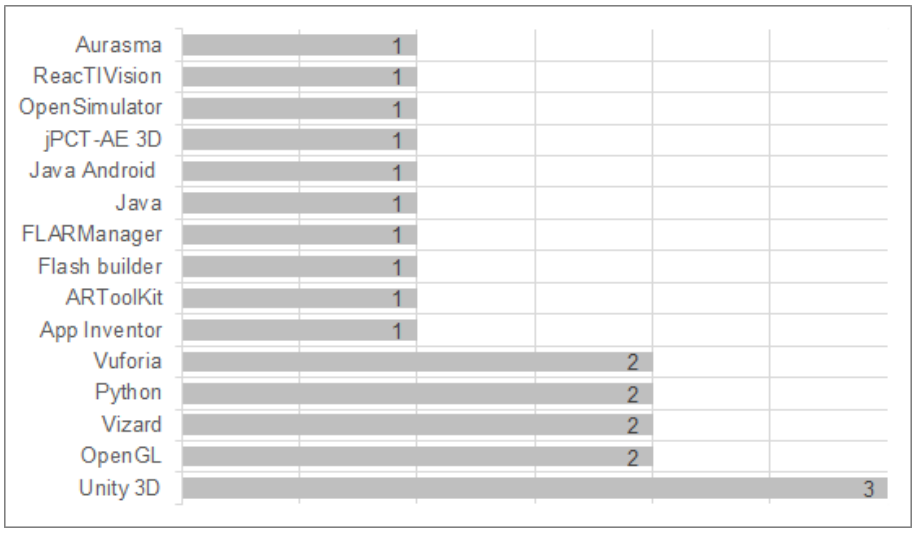
The studies approached programming concepts which are independent of programming languages, such as object-oriented programming (OOP) (S3, S8, S11, S12), logic structures (S5, S12, S14), data types (S3, S5, S12), loops (S5, S12, S14), arrays (S9, S12), basic programming logic (S1, S4), and so forth. Some primary studies are focused on specific programming languages and interfaces, as Prolog (S2), Arduino Scratch (S13), OpenGL (S6), and Python (S16).

RQ 4: What software development tools are used?

Despite part of the selected studies does not provide any information about the development tools that were used (S1, S3, S11, S12, S13), it was possible yet to identify several development tools for AR and VR applications, as Figure 21 illustrates. The studies have been adopting simulators, game engines, programming languages, development kits, and programming libraries.

We got the following arrangements of development tools: *(i)* OpenSimulator (S2); *(ii)* Unity 3D (S5); *(iii)* Unity 3D and ReacTIVision (S4); *(iv)* OpenGL and ARToolKit (S6); *(v)* App Inventor, Java Android, Vuforia, OpenGL, and jPCT-AE 3D Engine (S7); *(vi)* Vizard Simulator and Python (S8, S9); *(vii)* Java and Computer Graphics libraries (S10); *(viii)* Flash Builder, FLARManager, and LibsPark (S14); *(ix)* Unity 3D and Vuforia (S15); and *(x)* Aurasma (S16).

Figure 21 – Selected studies software development tools distribution



Source: Author

2.5 Final Remarks

This Chapter has presented the background for this Master's work. The teaching and learning of programming scenario was discussed, which allowed us to identify the current initiatives and challenges of programming education. We also introduced the fundamental concepts related to VR, AR, and MR.

Despite the increasing significance of programming skills among different audiences, there are some open issues to be addressed, since students and instructors still face problems and difficulties while teaching and learning programming. Specially in relation to m-learning applications, BBP, and VR, there are several features that could be used in benefit of the teaching and learning of programming.

The results of the mapping study we conducted reinforce the adoption of VR and AR as tools to support the visualization and practice of programming concepts, as well as to increase students' motivation and engagement. Results also showed some opening issues and opportunities for research in short term, mainly in relation to the use of mobile devices and immersive VR.

Considering the lack of mobile immersive VR applications and the benefits of BBP for learning programming, in the next Chapter we report on the creation of SSPOT-VR, a mobile immersive virtual world for the teaching and learning of programming to children and teenagers.

CHAPTER

3

A SPACE STATION FOR PROGRAMMING TRAINING IN VIRTUAL REALITY

In this chapter we present SSPOT–VR, an immersive virtual reality mobile application to support the teaching and learning of programming for children and teenagers. We developed SSPOT–VR considering that current educational VR applications focused on the teaching and learning of programming do not adopt mobile devices as hardware and software platform, as well as hardly often implement the learning benefits of immersion and presence brought by the usage of VR (MAKRANSKY; LILLEHOLT, 2018; AVELLAR; BARBOSA, 2019; SALAR *et al.*, 2020).

Even though the application can also be experienced at standard immersion level, using only the digital screen of mobiles devices as non-immersive displays, our virtual world was created to be experienced at a higher level of immersion using low-cost head-mounted displays (HMDs). We also aim at connecting students with SSPOT-VR using elements of storytelling and a programming method based on block-based programming principles. Our objective is to engage students in the virtual world narrative and in the programming task, in addition to assist and simplify the interaction within the virtual world.

In Section 3.1 we provide a detailed explanation of the application software we created, describing the context of SSPOT-VR and details on each level of the virtual world. In Section 3.2 we present and discuss the educational and software development aspects associated with the creation of SSPOT-VR. Finally, Section 3.3 concludes this chapter with our final remarks.

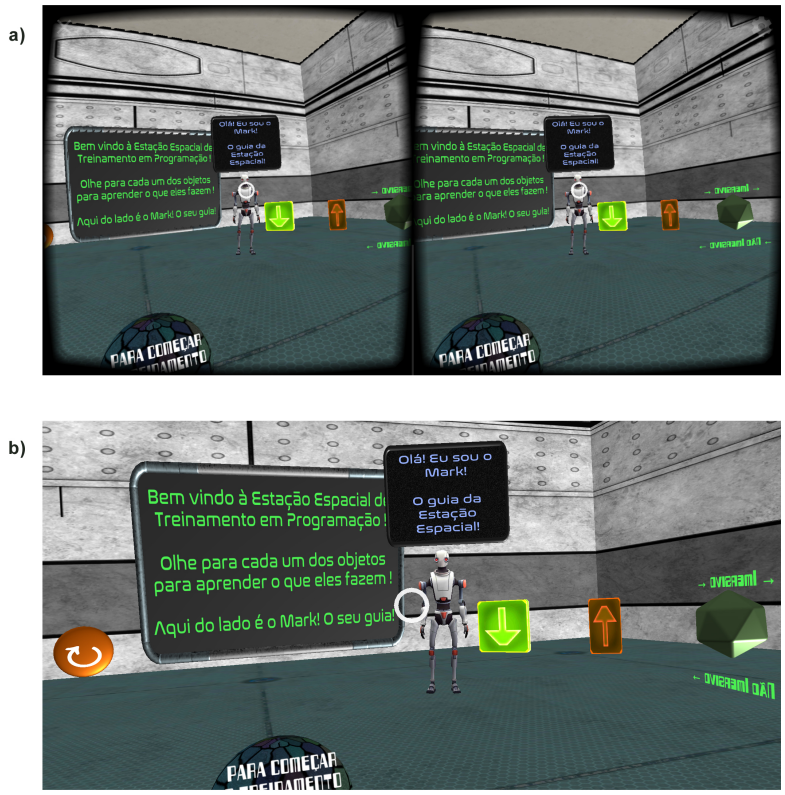
3.1 The SSPOT-VR Application

The Space Station for Programming Training in Virtual Reality or SSPOT-VR has been conceived as an immersive VR application for mobile devices with the purpose of assisting children and teenagers in the process of learning and practicing the concept of algorithm.

Although algorithm can be seen as a new term, constructing algorithms has been considered as one of the basis of the human intellect since its purpose is to describe, explain, and model the universe and its complex processes, being similar to reading, writing, and arithmetic. The algorithm concept can be applied in all areas and is clearly related to problem solving since it requires abstraction techniques for description and analysis of data and processes, as well as for the automation of solutions (SBC, 2019; WING, 2011).

Our application integrates methods for the teaching and learning of programming and the simulated experience of a digitally created world. Considering our scenario, a VR application is only practical when developed for the mobile platform, accordingly to it we rely on inbuilt sensors common to these devices and essential for VR mobile applications: accelerometers and gyroscopes. Despite we designed our mobile VR application to be experienced through stereoscopic view with a head-mounted display to provide a higher level of immersion (Figure 22 (a)), students may choose whether or not to use the HMD since the application can also be used as a full screen VR application, i.e., the magic window mode (Figure 22 (b)). A magic window allows users to view 360º content without a head-mounted display. The application renders a single full screen monoscopic view of the virtual world that is updated based on the device's orientation sensor[1].

Figure 22 – a) VR stereoscopic view of SSPOT-VR b) VR full screen view of SSPOT-VR



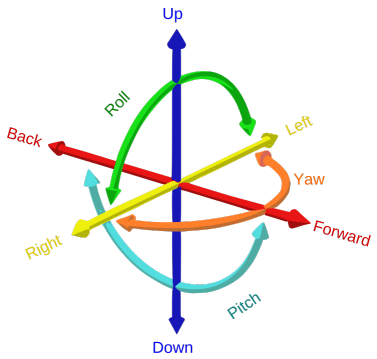
Source: Author

[1] https://developers.google.com/vr/develop/unity/guides/magic-window

In both usage settings, the movement of the mobile device is tracked through the sensors in order to provide to users 3 degrees of freedom (3-DoF). Tracking for 3DoF devices is done via the device's inertial measurement unit (IMU), a piece of hardware that measures and reports a body's angular rate and orientation. Although modern IMUs are accurate, tracking is still not perfectly accurate[2].

Degrees of freedom (DoF) define the number of directions it is possible to move an object through a 3D space, i.e., how the three different axes are being tracked. There are six degrees of freedom in total. 3-DoF correspond to the rotational movements around the x, y, and z axes, also known as pitch, yaw, and roll. This allows the application to track users head orientation, that indicates to where users are looking from a fixed viewpoint (Figure 23). 6-DoF correspond to the rotational movement around the three axes as well as translational movement along the x, y, and z axes, allowing tracking whether users have moved forward or backward, left or right, and up or down (JERALD, 2016).

Figure 23 – Degrees of freedom graph



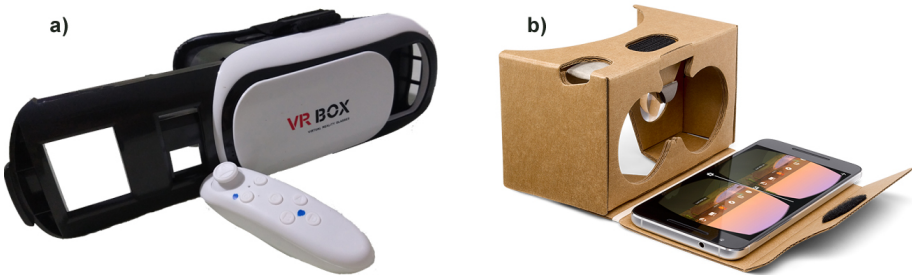
Source: Wikimedia Commons (2020)

The user interaction with the SSPOT-VR virtual world is straightforward. When there is no HMD employed in the usage, students are able to interact with the application through the tactile display of the mobile device. In the case of adopting an HMD to improve students' sense of immersion and presence, depending on the HMD model, the application might require a physical controller to interact with the virtual world, such as a wireless joystick (Figure 24 (a)). Besides the wireless joystick, the Google Cardboard, one of the most available HMD nowadays (Figure 24 (b)) has a built-in mechanism for performing actions designed as a lever that touches the screen with a capacitive pad when pressed by users.

In addition to the adoption of HMDs and high visual stimulation, our software application also benefits from sound stimulus, such as background music and sound effects for every interaction. We also included storytelling elements for keeping students focused and motivated whilst experiencing our virtual world. The realistic graphics, sound effects, and story simulate a typical science fiction environment, inspired by the stereotype commonly featured in sci-fi

[2] https://developers.google.com/vr/discover/fundamentals

Figure 24 – a) HMD with wireless joystick b) Google Cardboard HMD



Source: Author and Google (2020)

themed computer-animated movies and games. The sci-fi theme is adequate for SSPOT-VR in view of the fact that in addition to collaborating with the technological context that the application is part of, it is a popular theme among children and teenagers.

When the immersive experience initiates, students are presented to the setting that they are aboard a virtual space station specially created to teach concepts of programming to its visitors. At this space station lives a robot named Mark (Figure 25), which is the main character of our story. Mark plays the role of the virtual robot tutor of the students during their training in programming. In our plot, Mark's computer crashes and his memory has to be reset, making him forget how to move, preventing him to proceed with his duty as a tutor of the students during their journey aboard the space station. This conflict leaves the students responsible for fixing Mark, i.e., they need to reprogram Mark's movements in accordance with a secret movement algorithm which is presented to students as a sequence of textual instructions (FOG *et al.*, 2010). The students' task is to translate this sequence of text commands into a algorithm that Mark's computer can read. His computer is directed by principles of block-based programming and only accepts cubes to create algorithms.

The objective of SSPOT-VR is to joyful present the concept of algorithm. In an interactive way students create algorithms to program Mark's movements by assembling cubes that represent directions Mark can walk, with forward, right, and left movements. Students also need to define the algorithm begin and end, consequently defining Mark's start point and stop point. Students create the algorithm by using the cubes that symbolize the begin and end commands in addition to the assembling of the sequence of cubes with the directions that Mark must walk. After correctly inserting all the cubes into Mark's computer, students request the computer to run their algorithm, enabling them to observe the algorithm in action as Mark walks around accordingly. In other words, students use the secret movement sequence and the available cubes as the input to their algorithm and visualize in a interactive way the output of what they have programmed, as long as there are no compiling errors.

Before start interacting with cubes to learn and practice the concept of algorithm, students must go through an Introduction Level, in which they are welcomed to the space station and

Figure 25 – Mark, the tutor of the Space Station for Programming Training

Source: Author

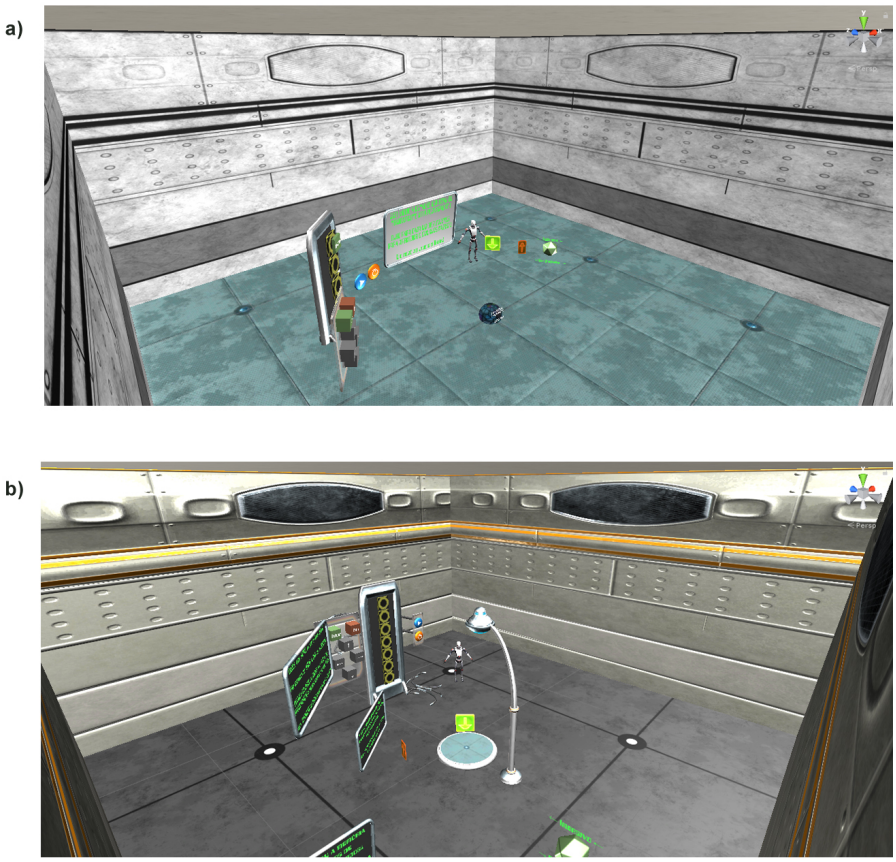
are also presented to what they will come across, such as the common objects of the space station scenario and Mark, our robot tutor. The Introduction Level is the first scene of the application while the First Challenge Level or Mark's memory reset level is the second one. As shown in Figure 26, students will experience the two different levels when using the SSPOT-VR application. Each level will be presented in detail next, together with more elaborate explanations on design, interaction, story, and programming method.

3.1.1 Introduction Level

Due to the innovative features of SSPOT-VR and considering that some students may have no prior experience using VR applications, we designed our VR space station for programming training to start with an Introduction Level as an effort for preventing students from feeling puzzled while experiencing SSPOT-VR for the first time. In order to students benefit from our virtual world, students have to understand the interaction methods and get comfortable with the special equipment. This includes the ability to use the gaze-based interaction and also the comprehension of the Head Mounted Displays' mechanisms, that in our case could be the Cardboard button that touches the screen with a lever or the usage of a wireless joystick. In addition to the introduction of usability of VR apps, it is at the Introduction Level that students have their first contact with the space station sci-fi theme and it is also where they get the first tips of the story to which they are now part of.

Gaze-based interaction works properly in almost every case and therefore is the most compatible and straightforward way to interact with VR apps, once it minimises physical effort by exploiting students' natural gaze movement. For each object that students point at, the gaze-

Figure 26 – a) Introduction Level overview b) First Challenge Level overview



Source: Author

based interaction draws a circular reticle pointer[3] in front of the object. If the students' gaze intersects a interactive object, the reticle automatically enlarge its size, as a sign of a possible interaction with that object. In front of students, as Figure 27 demonstrates, there is a welcome message to the space station, along with students' first instructions, that recommends students to look at each one of the objects to learn what they represent and what is possible to do with them.

Inside the Introduction Level of our virtual world, students are fixed at a position that enables them to observe the entire level, being only necessary to move their heads up, down, left, and right to change their perspective in order to view and interact with different parts of the level. VR apps need to manage changes in viewpoint and control input in a smooth and realistic approach to achieve immersion and keep the application usage comfortable. Building the sense of immersion is done through a careful design process with the combination of display, tracking, and audio features [4].

Despite the fact that gaze-based interaction is still susceptible to inaccuracies, such as unintentional activation, researchers have also demonstrated that interact with VR applications using gaze can increase levels of concentration, immersion, and presence (VINAYAGAMOOR-

[3] https://developers.google.com/vr/reference/unity/prefab/GvrReticlePointer
[4] https://developers.google.com/vr/discover/fundamentals

Figure 27 – a) The first state of the reticle pointer, when it is intersecting a non-interactive object, the instructive panel with the welcome message b) The second state of the reticle pointer, or the enlarged reticle, when it intersects an interactive object, as Mark, that says hello on its own instructive panel

Source: Author

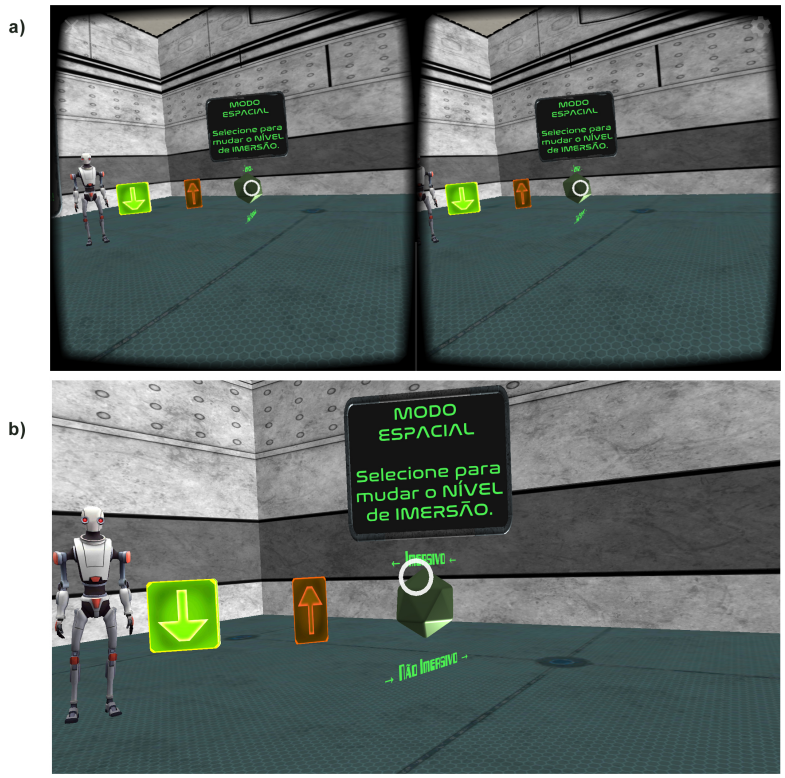
THY *et al.*, 2004; LEE *et al.*, 2016). On the other hand, screen touch or physical button press can avoid this problem considering that manual confirmation combined with gaze interaction is a good fit (ZHAI; MORIMOTO; IHDE, 1999; PFEUFFER *et al.*, 2017). Besides the primary difference between VR and traditional apps, that is with VR, the user is immersed within a virtual world; the gaze-based interaction, usage of HMD, and manual confirmation are also not common. Hence, our goal was to create an uncomplicated Introduction Level, to assist students in using a VR application and to present some of the background and setting of SSPOT-VR.

The intent of the Introduction Level is to drive students to take a closer look at each one of the key objects of the space station. The objects are in front of them, on their right and on their left. Students need to put the reticle pointer on top of each one of the objects to learn about them. When the reticle intersects an object, above this object an instructive panel appears together with the associated sound of that object that is played. We created the interaction with the objects of the Introduction Level based on the occurrence of two events. First, when the reticle pointer intersects an object, the instructive panel is displayed; and then, when the reticle pointer no longer intersects that object, the panel is hidden. In this case, the action is triggered automatically, no manual confirmation is required such as button press or screen touch.

On students' right, as shown in Figure 28, there is an object that we named space mode.

This object acts as a button to switch between stereoscopic rendering and magic window mode. In the case of interaction with the space mode button, manual confirmation is required. The option to switch between stereoscopic rendering and magic window mode is available at both levels for students change it at any time while using the application.

Figure 28 – Space mode button a) Stereoscopic rendering b) Magic window mode



Source: Author

Although students are completely free to change their perspective within the virtual world (considering the 3DoF), they need to realize that at the Introduction Level they will always be in a fixed position. This also means that movement inside the virtual world is based on predetermined points and teleportation. Teleportation is a locomotion mechanism that allows students to move around in VR environments with minimal discomfort. With teleportation, students points the reticle pointer at a location where they need to move to, and once the teleportation action is started, they are rapidly transitioned to that location[5]. That being so, still on the students' right, there is the programming platform button (Figure 29 (a)), used to place students in position to program Mark's computer, and also the teleportation arrow (Figure 29 (b)), that represents the predetermined points that students are able to teleport.

On the other hand, on the left of the Introduction Level are the objects that are part of the programming method that we developed from the principles of block-based programming, starting with the collection of cubes that students will use to program, as illustrated in Figure

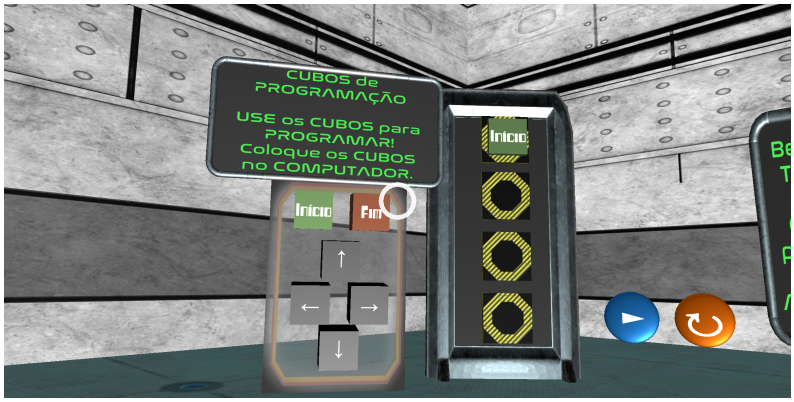
[5] https://developers.google.com/vr/elements/teleportation

Figure 29 – a) Programming platform button b) Teleportation arrow

Source: Author

30. At this moment, students do not know what the cubes represent and how they can be used, but the arrows printed on the cubes are commonly used to indicate movement which present hints of what they will be doing in the first challenge. In addition, the instructive panel of the collection of programming cubes is informing students that the cubes are used for programming and that they must be put in Mark's computer, located right next to the collection of cubes. Further explanations on the created programming method using cubes and Mark's computer are given in Subsection 3.1.2.

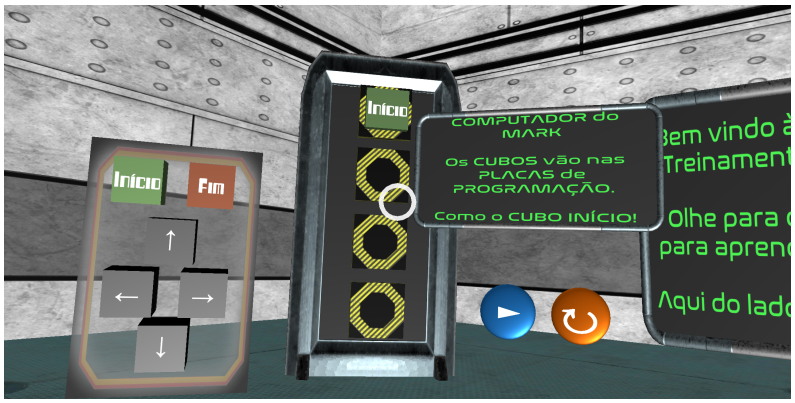
Figure 30 – The collection of programming cubes to be used during the programming challenge



Source: Author

As mentioned, right on the side of the collection of programming cubes, there is Mark's computer. Mark's computer is the virtual computer students are going to program using cubes. As shown in Figure 31, we put a green *begin* cube attached to the first one of the four programming boards of Mark's computer to demonstrate to students how the programming cubes should be used. More information on how to effectively take a cube and put it into one of the programming boards are given to students only during the first challenge.

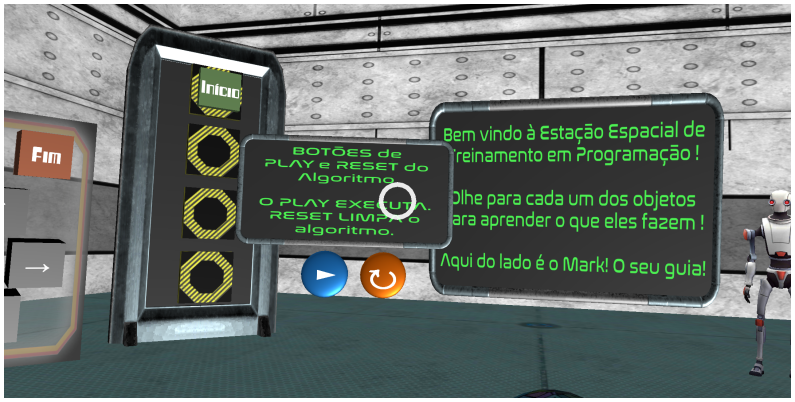
Figure 31 – Mark's computer with an example of a cube put into one of its programming boards



Source: Author

Finishing the presentation of the objects that are part of the programming method developed for the SSPOT-VR (Figure 32), there are the play and reset buttons. Similarly most integrated development environments (IDE), we adopted the play sign for our run button. Our intention is to emphasize that is essential to somehow run the algorithm in order to receive its output. Although a button that resets the algorithm is not common to IDEs, we adopted a reset button whose function is to remove all the cubes attached to the programming boards and clean up compilation errors, resetting Mark's computer to receive a new algorithm.

Figure 32 – Play and reset buttons, to run the algorithm and to reset the computer, cleaning up previously written algorithms on the programming boards



Source: Author

The Introduction Level, in addition to being an environment for students to learn about how VR apps work, is an environment in which students can see how our sci-fi space station looks like and obtain information on how to make use of its features. We present key objects and features to students, such as how to switch between stereoscopic rendering and magic window mode, how to move around, and how they can go to another level from where they are. Once students feel more comfortable with our virtual world, the programming training can be started using the button on the floor of the Introduction Level, as can be seen in Figure 33. Thus we expect that understanding how SSPOT-VR works, students can feel part of the story and effectively benefit from what the application can provide. After exploring the Introduction Level, students are ready to the first challenge, which is presented next.

Figure 33 – Button to teleport to the first challenge

Source: Author

3.1.2 First Challenge Level

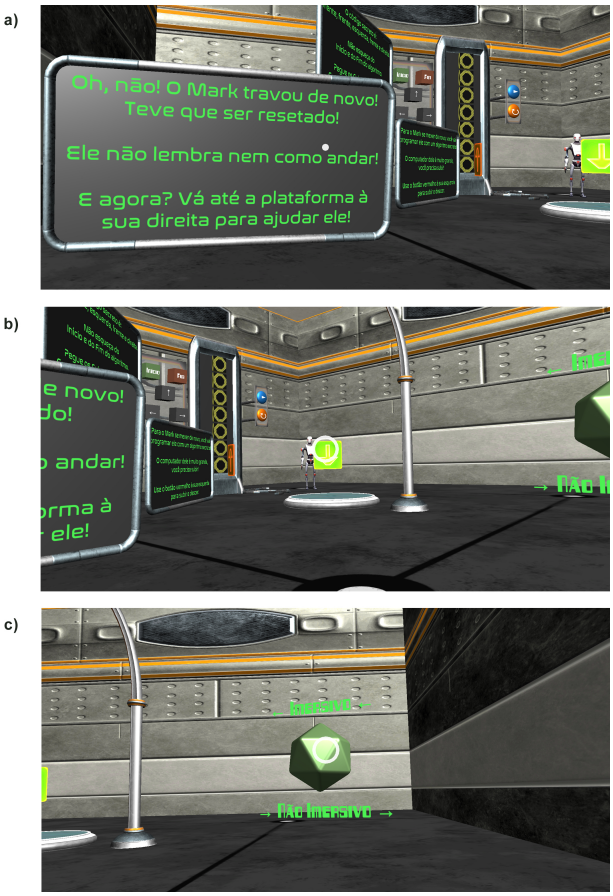
Once students arrive at the second of our application, the First Challenge Level or Mark's memory reset level, it is important that they start exploring different perspectives of the level, following the instructions available on instructive panels placed around the level to figure out what kind of challenge they have been tasked with. As illustrated in Figure 34, after using the button on the floor of the Introduction Level to teleport to the first challenge, students have an overall viewpoint of this level.

As seen in Figure 34 (a), the instructive panel right on front of students presents the initial part of the story. We tell to students that Mark's computer crashed and had to be reset, which ruined some parts of Mark's memory. In other words, the reset caused the deletion of Mark's movement algorithm from his memory, making it necessary to somehow fix it or rewrite it. We chose the crash and freeze problem because it is an usual problem nowadays for devices and machines that are based on software to operate.

The first direct instruction is also provided to students, informing that to help Mark is necessary to go to the platform on the right. The teleportation arrow to the programming platform

is illustrated in Figure 34 (b). From this point forward, students will keep receiving instructions to tasks that should be accomplished in order to carry out the challenge. Further on students' right, as shown in Figure 34 (c), there is also a space mode button in the first challenge, allowing students to switch between stereoscopic rendering and magic window mode.

Figure 34 – a) In the front point of view, the first instructive panel b) At students' right, the teleportation arrow to the programming platform c) Further right, the space mode button



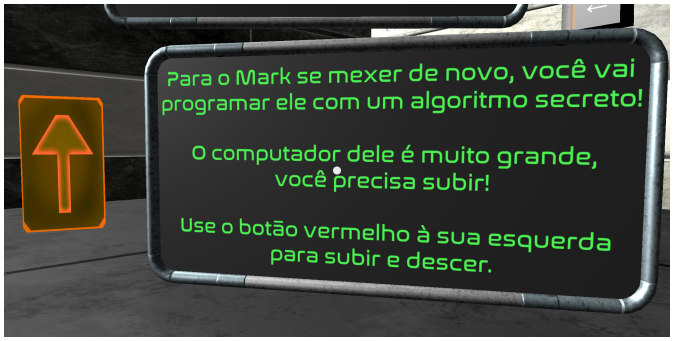
Source: Author

The communication between the virtual world and students is done through the instructive panels with the purpose of acting positively in the immersion of our story. The provided instructions are within the plot we created, no instructions mention actions in the real world with real devices, such as instructions to press buttons on the HMDs, joysticks, or touch the device screen. Moreover, we use only a few direct instructions, such as to go to a certain location, use a specific button of the virtual world, and so on.

After students used the teleportation arrow for the first time and got the programming platform, another instructive panel describes to students the current state of the plot and how to proceed with the challenge. We inform students that they need to program Mark based on a secret code in order to fix his memory problem. In addition to the secret code, that has not yet been revealed, students need to move one last time. This time, students are required to go up.

While on the programming platform either up or down, students are completely free to return to the starting position, move up and down with the platform, and observe different perspectives of the level from their position. We also inform through the instructive panel that Mark's computer is huge, therefore it is necessary to go up to be able to program it. This is due to the need to place students at the ideal position to carry out the programming activity. To continue, students must interact with the programming platform button, which functioning is similar to an elevator button. It helps to move the programming platform up and down, as Figure 35 illustrates.

Figure 35 – Instructions for when the programming platform is down



Source: Author

Considering that the programming activity inside the SSPOT-VR is based on assembling cubes on top of each other, we created the programming platform to make the act of programming more comfortable. The angle of movement that students' neck needs to make during the programming activity is minimized when the programming platform is up[6].

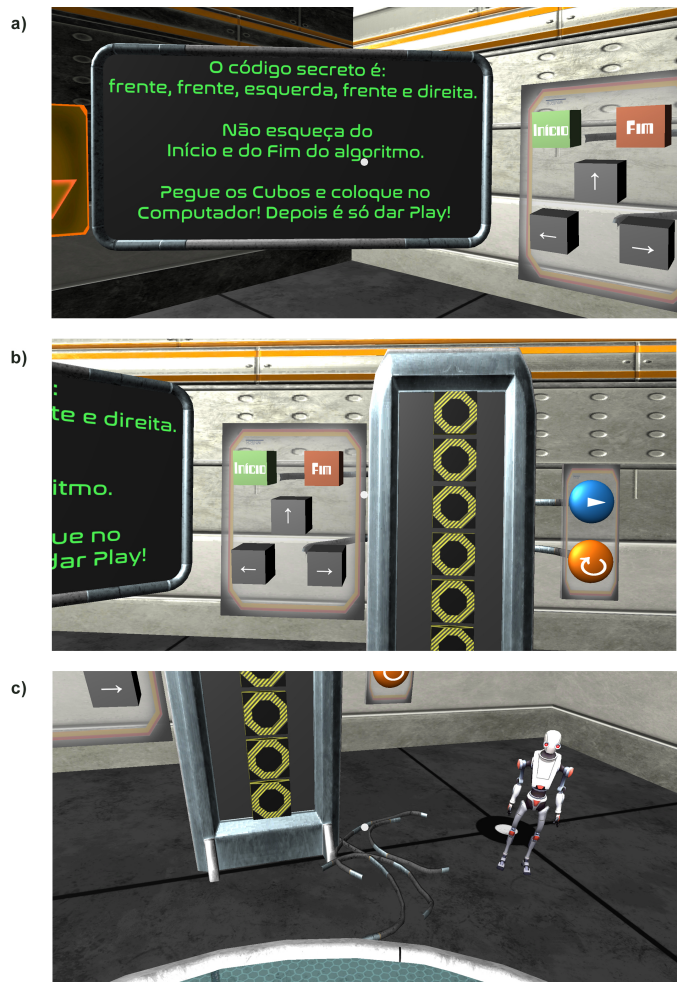
When students go up with the programming platform, a new instructive panel provides the secret code to repair Mark's memory and basic instructions on how to program Mark's computer, as shown in Figure 36 (a). The secret code is informed in text, that is the sequence of: front, front, left, front, and right. In this activity we considered algorithm as a description of a process for solving a problem or accomplishing some goal (SBC, 2019).

To solve the challenge, or fix Mark's memory reset, students must program Mark's computer with the cubes at least once, accordingly to the secret code. At the moment the objects presented to students at the Introduction Level, the cube collection, Mark's computer, and the play and reset buttons, are in the front view of the students and ready to be effectively used (Figure 36 (b)). We can also see Mark waiting to be fixed on the right of his computer (Figure 36 (c)).

Simple instructions on how to program are also provided to students. We request them to take the cubes, put them on the computer, and press the play button. Take a cube and put it in the computer must be intuitive to students, using the combination of gaze-based interaction

[6] https://developers.google.com/vr/develop/experimental/6dof-design-guidelines

Figure 36 – a) Secret code and basic instructions on how to program Mark's computer b) Programming cubes, Mark's computer, and the play and reset button c) Mark waiting to be fixed



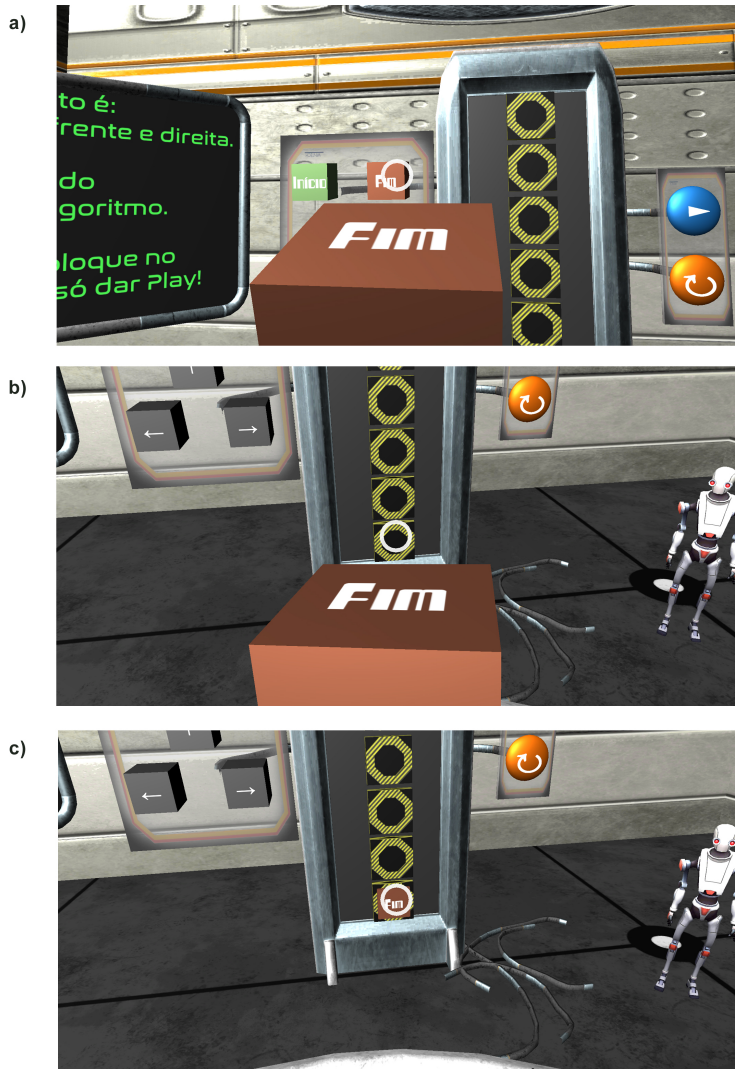
Source: Author

with manual confirmation, i.e., gaze in addition to the touch on the screen, HMD or joystick button press. So far, students do not know Mark will walk accordingly to the cube-algorithm that is being executed in his computer. However, in order to solve his memory reset problem, it is needed to create an algorithm with cubes identical to the secret code; otherwise Mark will walk but the memory problem will persist. This is important to show students that although the algorithm can be executed and generate an output, it is not yet the algorithm that represents the solution to the problem.

The interaction with the programming cubes is based on an hold and release approach. To hold and release the cubes students only need to use the gaze-based interaction and manual confirmation. Once students have positioned the gaze over the cube and performed manual confirmation, the cube will remain in the student's hands, who will hold it until s/he decides to release it or put it in the computer (Figure 37 (a)). In both cases, it is only necessary to perform the confirmation again, which means touching the screen or pressing the HMD or joystick button. To release the cube, confirmation is required again when the gaze does not intersects any

interactive object. To put it in the computer, it is necessary to position the cube on the computer's programming board and then perform the confirmation again (Figure 37 (b) and (c)).

Figure 37 – a) Student holding the end cube b) Positioning the cube on a programming board c) End cube in the last programming board



Source: Author

As mentioned, the programming activity students perform within the virtual world is based on principles of block-based programming and is characterized by assembling two types of cubes. One represents Mark's movements and the other represents the begin and end of the algorithm. In addition to turning straightforward the interaction for the programming activity in mobile VR applications, visual programming languages are recommended for different authors for children and teenagers to learn programming concepts (WEINTROP, 2019; SBC, 2019; RAABE; BRACKMANN; CAMPOS, 2018; CAMPOS; DIAS, 2020).

In addition, according to the SBC guidelines for teaching computing in basic education of the *Sociedade Brasileira de Computação*, more specifically, the guidelines focused on teaching computational thinking, the use of a visual programming language is suitable for the target

audience of SSPOT-VR. Therefore, with SSPOT-VR it is possible to exercise the skills of (SBC, 2019): (i) understanding the definition of a problem as a relationship between input and output; (ii) usage of visual programming languages to describe solutions to problems involving basic instructions, and (iii) relate programs described in visual language to texts and vice versa.

Although there is a specific algorithm to solve Mark's memory problem, students can freely program Mark's computer using the cubes. However, the algorithm will only work if there are no compilation errors. We have defined three conditions for the compiler of Mark's computer to execute a cube algorithm. The first condition to run a cube algorithm in Mark's computer is that all programming boards must contain a programming cube.

As illustrated in Figure 38, when students do not fill in all the programming boards and press the play button, Mark's computer changes its background to a reddish color to warn students that something is wrong. Besides changing the computer background color, we inform students what sort of error has occurred with a message on a panel positioned above Mark. In this case, we inform students to fill in all the programming boards of Mark's computer with programming cubes.

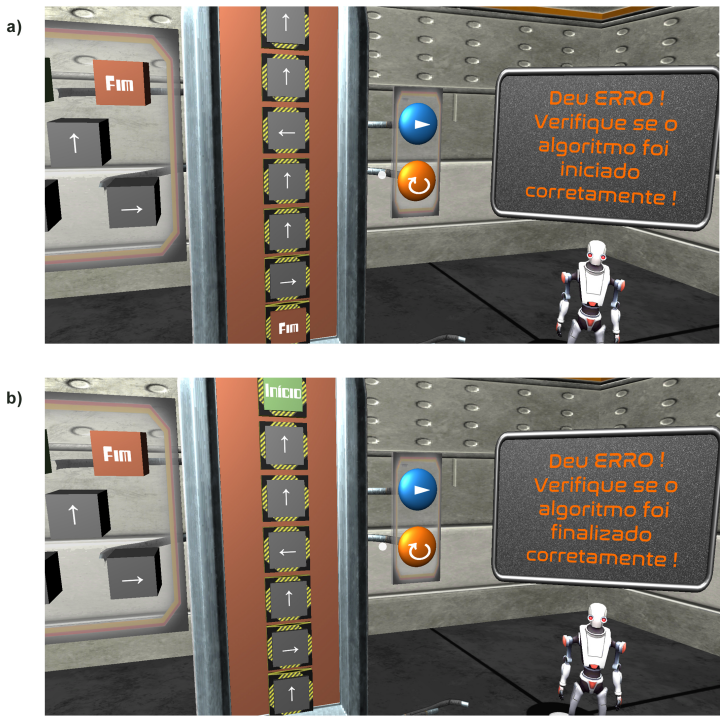
Figure 38 – Error status for not filling all programming boards with cubes

Source: Author

Secondly, it is necessary to properly declare the begin and end of the algorithm. We require students to start and finish the algorithm with the respective begin and end cubes, putting the begin cube in the first programming board and the end cube in the last. If students put another cube on the first programming board and hit the play button, the panel above Mark reminds them to verify if the algorithm was correctly initiated (Figure 39 (a)). The same is true if students place another cube on the last programming board other than the end cube (Figure 39 (b)).

Lastly in the conditions for compiling algorithms on Mark's computer, students can only use the begin cube and end cube in their proper places, in the beginning and in the end of the algorithm. As can be seen in Figure 40, it is informed to students that between the start and end cubes there should be only arrow cubes. Our intention is to make students realize that there are different types of cubes to be used when programming and that each type of cube has specific

Figure 39 – a) Error status for incorrectly initializing the algorithm b) Error status for incorrectly finishing the algorithm



Source: Author

functions and characteristics, such as the command that the cube represents and the place that it should be used in an algorithm.

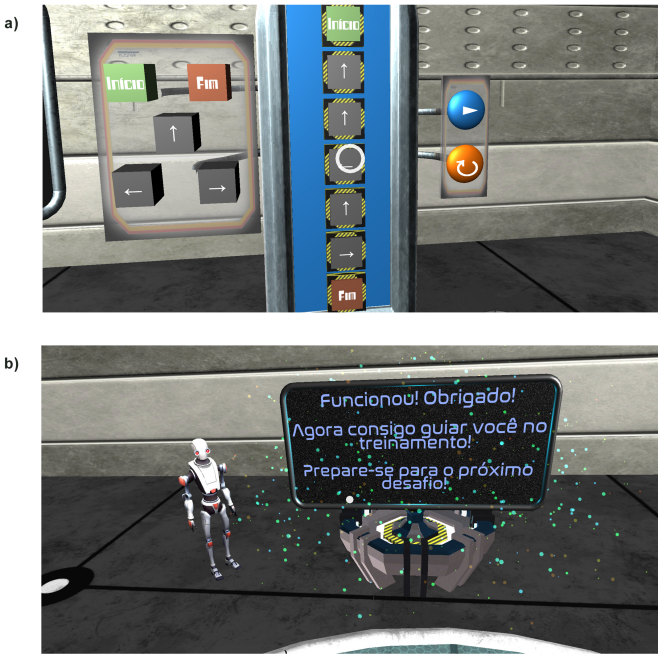
Figure 40 – Error status for incorrectly usage of begin or end cubes

Source: Author

After pressing the play button, the compilation will succeed only if the algorithm created by the students meets the three compiling conditions: (i) fill all of the programming boards with a cube; (ii) start and finish the algorithm with the correct cubes, and (iii) use begin and end cubes only in the right positions of the algorithm. To represent the success on running a algorithm, Mark's computer changes its color to blue (Figure 41 (a)). When Mark walks accordingly to the secret code, he will stop next to a celebration panel, where student success in the challenge

is announced. In this message, Mark thanks the students for their help, informs them that the algorithm worked, and that students should prepare for the next challenge (Figure 41 (b)). We also recorded a video[7] to better present the visual and sound present in our application.

Figure 41 – a) Mark's computer displaying the success status b) Fanfare panel that concludes the First Challenge Level



Source: Author

After students experience all the situations we described along this section, the story ends with Mark celebrating with the students. Although, it does not mean that students cannot continue exploring the two levels of SSPOT-VR. The idea of our application is to foster exploration and creativity, so we provide a loop scenario that allows students to explore as much as they want.

As mentioned, the programming content covered at this level, which includes the concept of algorithm and the usage of a visual programming language, is based on guidelines for teaching programming to children and teenagers (SBC, 2019; RAABE; BRACKMANN; CAMPOS, 2018). Besides the concept of algorithm, which is one of the first contents that should be addressed with children and teenagers, according to the works of SBC (2019) and Raabe, Brackmann and Campos (2018), the authors also provide a complete planning to the teaching and learning of different introductory contents of programming, as well as intermediate and advanced contents, for students of primary and secondary schools.

In addition, even though we used a game engine together with a software development kit for VR to build the SSPOT-VR, construct a virtual world is not simple, it is necessary to design and code all that students will visualize and interact. Moreover, most assets that we used had to

[7] https://youtu.be/LXb8WFqUVdw

be modified to suit what we needed for an application that teaches programming to children and teenagers. In this context, we created the design of two more levels of the SSPOT-VR application: the Second Challenge Level and the Third Challenge Level. Next, we present the design of these two new levels of SSPOT-VR that approaches two new concepts of programming.

3.1.3 Second and Third Challenge Levels

The Second Challenge Level and Third Challenge Level are based on the First Challenge Level narrative. Both of them avail the programming activity that was already proposed, which is to create an algorithm with blocks to make Mark walk. However, we approach new programming concepts to be practiced by students. It is also important to mention that we did not code the second and third challenge levels, i.e., we only created the scene design of the virtual world, therefore, there is no interaction or behavior programmed or scripted with the environment or objects. This means that, the evaluations conducted with SSPOT-VR, presented in Chapter 4, only include the usage of the Introduction Level and the First Challenge Level, which are complete for usage and fully functional. Although, in the future, we plan to evaluate the SSPOT-VR with the three different challenges.

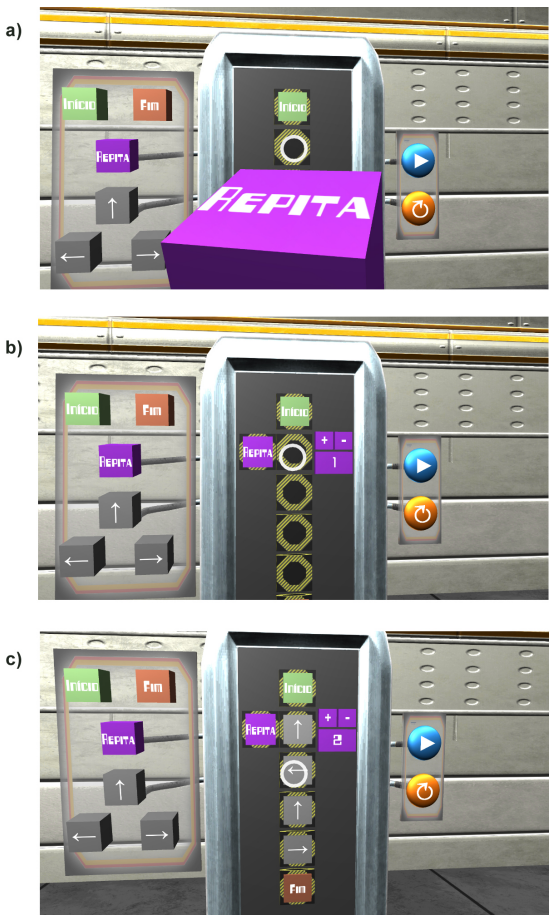
Following the programming content covered by the programming teaching guidelines of SBC (2019) and the reference curriculum of Raabe, Brackmann and Campos (2018), and considering that the concept of programming addressed in the first challenge of SSPOT-VR is the concept of algorithm, the next concepts that we address are the concept of repetition structures with fixed number of iterations and the concept of variable. As figure 42 reveals, we added a new cube to the collection of available cubes of the second challenge of SSPOT-VR, the repeat cube.

Figure 42 – The new repeat cube among the collection of cubes at the second level

Source: Author

The goal of the second challenge of SSPOT-VR is therefore to present to students the functioning of an algorithm that uses a repetition structure with a fixed number of iterations using the repeat cube, as Figure 43 displays. In the programming method we have adapted, the repeat cube represents a structure that repeats an instruction of an algorithm for a fixed number of times.

Figure 43 – a) User holding the repeat cube b) Repeat cube re-positioned after put into Mark's computer c) The same algorithm approached at the first challenge using the repeat cube



Source: Author

After users have hold the repeat cube and put it in Mark's computer, the cube behavior is different from what students were used to. When placed on any programming board of the computer, the repeat cube is re-positioned to a programming board on the left, then, an empty programming board appears in the center, and on the right there is a numerical counter. The counter is an exclusively repeat cube feature that allows students to increase and decrease the number of times that the instruction that goes on the center programming board will be executed on that algorithm.

As we mentioned, in the second challenge of SSPOT-VR, students create the same algorithm that was created in the first challenge, however, using a repetition structure. This implies in the use of one less programming board, i.e., the same algorithm that produces the same result can be created with one less instruction when using a repetition structure.

In the third challenge, students must create the same algorithm again, but with more modifications. In addition to using the repeat command, students will now have to define the number of iterations of the repeat cube using a variable and instantiating the value of this variable. In the third challenge, therefore, we introduce the concept of variable and how to use

algorithms to modify the values of variables. There is, consequently, another new cube, as Figure 44 illustrates.

Figure 44 – The new var cube among the collection of cubes at the third level

Source: Author

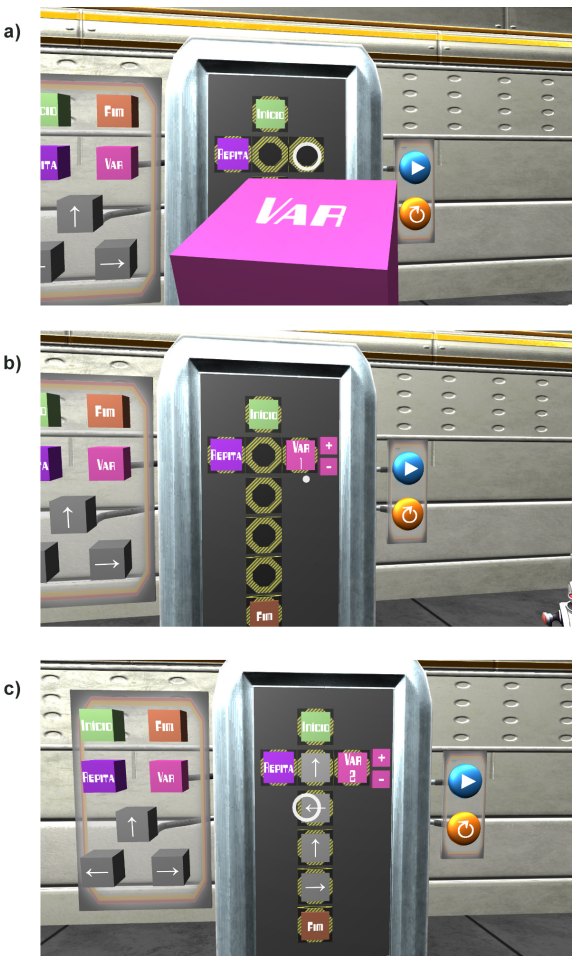
As Figure 45 demonstrates, after students put the repeat cube in Mark's computer, it is re-positioned to the left, and this time, there are two programming boards to the right that are empty for students to fill them out. Similarly to the second challenge, the student must place the instruction that will be repeated on the center's programming board and on the right programming board it is necessary to define the number of repetitions that this instruction will be carried out by the repeat cube. The numerical counter is defined this time by the value of a variable, which must be placed on the rightmost programming board and its value selected from the increase and decrease buttons.

The objective of the third challenge, in addition to presenting the concept of variable and using algorithms to modify its values is also to present that an algorithm can be a generic solution to a problem and variables can be used to describe generic solutions. Generic solutions are also part of the works of SBC (2019) and Raabe, Brackmann and Campos (2018).

The purpose of presenting the scenes of the second and third levels is to demonstrate that SSPOT-VR is able to address different programming content combining VR, mobile devices, block-based programming, and storytelling. As we mentioned, the second challenge and the third challenge did not go through the coding or scripting phase during this work. Only their scene designs have been built and they were not part of the SSPOT-VR evaluation. This perspective is also illustrated in Chart 4, which presents an overview of the SSPOT-VR application.

Examining the chart, it is possible to note that the Introduction Level and the First Challenge Level are fully functional and were evaluated from the point of view of user acceptance and use and usability. The programming contents approached in the first challenge of SSPOT-VR are the concept of algorithm and the usage of a visual programming language. According to SBC (2019) and (RAABE; BRACKMANN; CAMPOS, 2018), the SSPOT-VR application approaches programming contents appropriate for students from 6 years old. However, considering that SSPOT-VR adopts mobile devices and a block-based-programming language, which is a type of

Figure 45 – a) User holding the var cube b) Repeat cube and var cube put into Mark's computer c) The same algorithm approached at the first challenge using the repeat and var cube



Source: Author

Chart 4 – Summary of SSPOT-VR levels

ID	Level Name	Programming content	Target audience		Development stage	Evaluation
			SBC (2019)	Raabe, Brackmann, and Campos (2018)		
1	Introduction	None. Only instructions to use the virtual world	N/A	N/A	Fully functional	User acceptance and use and usability
2	First Challenge	Algorithm	6 years old	6 years old	Fully functional	User acceptance and use and usability
		Visual programming language *	11 years old	11 years old		
3	Second Challenge	Repetition structure	9 years old	7 years old	Only scene design	Future work
4	Third Challenge	Variable	11 years old	8 years old	Only scene design	Future work
5	Fourth Challenge	Decision structure	8 years old	9 years old	Future work	Future work
6	Fifth Challenge	Array	12 years old	9 years old	Future work	Future work

visual programming language, our virtual world is recommended to be used for students who have at least 11 years old. Despite that, students younger than 11 years old are not prevented from using SSPOT-VR, as it was the case during the evaluation of user acceptance and use, that we conducted with SSPOT-VR and presented in Chapter 4.

In contrast with the Introduction Level and the First Challenge Level, the Second and

Third Challenge Levels have only their design scenes created, that have been presented throughout this section. Evaluation of the second and third challenge are future work, as well as the planning, design, and development of the Fourth Challenge Level and the Fifth Challenge Level, that complete the SSPOT-VR application. The fourth challenge approaches the concept of decision structure, while the fifth approaches the concept of array.

It is thereby possible to sequentially address some of the fundamental concepts of programming in SSPOT-VR which are in accordance with the sequence of educational stage of primary and secondary school students, considering the work of Raabe, Brackmann and Campos (2018). We may also notice that not always both documents that we have adopted to guide the educational aspect of SSPOT-VR address the same content for the same age of children and teenagers. This is one of the difficulties with the teaching and learning of programming, as discussed in Section 2.1.1.

Next, we present the details of the educational and technical aspects associated with the creation of SSPOT-VR.

3.2 Educational and Technical Aspects

Although we adopted available documents for guiding the creation of SSPOT-VR, it required customization and adaptation of what the literature presents about methods of teaching and learning of programming and development of VR applications. The purpose of this section is to present the educational and technical aspects associated with the creation of SSPOT-VR. In the following sections, we explain how we used the the works of SBC (2019) and Raabe, Brackmann and Campos (2018), and the development documentation[8,9,10] in order to make our design and project decisions.

Explaining the design and development of SSPOT-VR is important since most of the efforts in the field of VR applications are based on trial and error, driven by intuition and common-sense rather than being underpinned by methods, models, or frameworks (DALGARNO; LEE, 2010). Also considering the gap of mobile educational applications in VR to support the teaching and learning of programming, as well as the lack of studies that provide directions for the creation of such applications, our intention is to enable researchers and instructors who are interested in using mobile learning and VR to use SSPOT-VR as inspiration or model for creating their own applications (AVELLAR; BARBOSA, 2019).

[8] https://developers.google.com/vr/develop & https://developers.google.com/cardboard/develop
[9] https://docs.unity3d.com/Manual/VROverview & https://learn.unity.com/tutorial/vr-best-practice
[10] https://developer.oculus.com/learn

3.2.1 Teaching and Learning of Programming

Considering that the teaching and learning programming is a complex task both to learn and to teach, even when it comes to children and teenagers, it is necessary to design the application design to specifically meet the needs of a target audience (ROBINS; ROUNTREE; ROUNTREE, 2003; PITEIRA; HADDAD, 2011). It is also necessary to design the programming content and the method in which this content will be presented based on teaching guidelines, in order to offer quality, practical, and flexible learning opportunities (SBC, 2019; RAABE; BRACKMANN; CAMPOS, 2018).

In this context, we identified a lack of mobile immersive VR apps focused on the teaching and learning of programming. We also identified that mobile learning applications, block-based programming principles, storytelling elements, and VR have been separately supporting the teaching and learning of programming (AVELLAR; BARBOSA, 2019). Therefore, we investigate whether a mobile immersive VR application for teaching programming to children and teenagers using BBP principles and storytelling elements would be used as a support mechanism for teaching programming to children and teenagers.

Next, we discuss how we defined the target audience for SSPOT-VR. After defining the target audience, it is possible to select the programming content, as well as the teaching and learning method.

Target Audience

We chose children and teenagers as the target audience for SSPOT-VR considering that the integration of mobile learning, immersive VR, block-based programming principles, and storytelling elements would reach its maximum potential with this target audience (HAMILTON; WEISS, 2005; WEINTROP; WILENSKY, 2015; VINCUR *et al.*, 2017; SEGURA *et al.*, 2020). We also consider that most VR applications that support teaching and learning of programming obtained through our mapping study are for students of introductory courses of Computer Science undergraduate programs, revealing that there are possibilities for new research when the target audience is children and teenagers (AVELLAR; BARBOSA, 2019).

Bearing in mind that it is possible to start teaching programming for 7 years old students using unplugged activities, i.e., not using computers or any electronic device; and considering the technologies and devices we have adopted to create the SSPOT-VR application, our virtual world is recommended to be used for students that have at least 11 years old. At this age, students are more prepared to manipulate mobile devices and software applications (SBC, 2019).

The definition of the target audience is essential for the creation of the application, as it defines: (i) the specific needs and requirements of the virtual world; (ii) the complexity of the programming content; (iii) the method through content is presented; (iv) the visual of the virtual world, and (v) the user's interactions.

Given that SSPOT-VR was created to be used mainly by students from primary and secondary schools, we adopted the programming teaching guidelines of proposed by SBC (2019) as the main document to guide the definition of programming content and teaching method. The reference curriculum proposed by Raabe, Brackmann and Campos (2018) was also considered. Both documents points out programming contents to be taught to all grades of primary and secondary school.

After the definition of the target audience, different issues arise and the curricular guidelines and reference curriculum should be used to settle them, indicating what to teach and learn, how to teach and learn, and what pedagogical practices can be used to inspire new ways of teaching and learning. We are now able to select the programming content to be covered in the application.

Programming Content

When defining the programming content, it is important to note that virtual worlds should have a predominance of objects with intuitive and similar usage to the real world, together with the presentation of objective instructions and short textual information. It is also important to try to motivate and engage students in experiencing the virtual world using playful and visual appealing content.

In virtual worlds, everything that users see and interact have to be planned and developed, including visual representations of programming concepts and programming activities. It may happen that the more complex the programming content, the more complex it is to transform this content into something viable to be approached in VR. In addition to that, there are evidence that applications that use virtual reality to support programming teaching and learning usually address introductory programming content. However, nothing prevents advanced concepts from being addressed (AVELLAR; BARBOSA, 2019).

In view of this, we selected the concept of algorithm to teach to students using the SSPOT-VR application. Algorithm is an essential concept for programming, both SBC (2019) and Raabe, Brackmann and Campos (2018) approach this concept in an unplugged way in the early years of primary school. Students need to understand the processes of problem solving and realize the importance of being able to describe solutions in the form of an algorithm.

The concept of algorithm is adequate to be addressed in SSPOT-VR as it can be associated in different ways with the students' daily lives, in addition to facilitating the learning and practice of other programming concepts. When students are experiencing SSPOT-VR, they must translate an algorithm sequentially described in Portuguese to an algorithm in a block-based programming language.

Besides the learning of the concept of algorithm, we make it possible to exercise the skills of (RAABE; BRACKMANN; CAMPOS, 2018; SBC, 2019): (i) understanding the definition of a

problem as a relationship between input and output; (ii) usage of visual programming languages to describe solutions to problems involving basic instructions, and (iii) relate programs described in visual language to texts and vice versa. Considering the sequence of programming concepts established by Raabe, Brackmann and Campos (2018) and SBC (2019), the programming content covered by SSPOT-VR is a combination of contents from the early years of primary school.

Based on SBC (2019) and Raabe, Brackmann and Campos (2018), it is possible to approach more advanced programming concepts with 11 year old students. However, the authors recommend that electronic devices should be used for educational purposes only by students of at least 11 years old. Therefore, SSPOT-VR contemplates programming content prior to what is specified for 11 year old students, such as the concept of algorithm.

The objective of SSPOT-VR is to firstly address fundamental programming concepts in order to be able to coherently advance to intermediate and advanced programming concepts. After that the concept of algorithm is learned, it is possible to cover repetition structures, variables, and so forth. The different programming concepts that SSPOT-VR addresses are divided into different levels that are presented in Section 3.1.

After establishing the target audience and the programming content, we have to select an appropriate teaching and learning method.

Teaching and Learning Method

As we created SSPOT-VR to be actively experienced by students we chose to teach the concept of algorithm through the practice of creating algorithms with a block-based programming language. This activity is associated to the participation of students in a narrative. Our goal is to help students to learn solving problems by trial and error methods, exploring, and experimenting. This can be done through analogies, metaphors, and storytelling, which guide students through participation in narratives with programming concepts associated. An analogy can benefit a student's understanding of a concept by making abstract information concrete.

BBP is a visual programming approach that can be more immersive when combinated with VR, in addition to being practical and interchangeable for textual programming. The interaction with programming blocks in VR stimulates creativity and discovery. BBP languages make it possible to avoid some of the frustrations that arise when learning syntax and allow the student to focus on the idea of solving the problem (MOHAMAD *et al.*, 2011; WEINTROP; WILENSKY, 2015; BAU *et al.*, 2017).

At the same time, storytelling is about using narratives to engage students in the process of learning. It creates a participatory and immersive experience that allows students to enjoy learning as a dynamic and entertaining method. Besides that, stories and narratives are the way we store information in the brain. When students are emotionally involved with the story, it can help them develop a positive attitude toward the learning process. Even students with low

motivation and weak academic skills are more likely to participate in the context of storytelling (HAMILTON; WEISS, 2005; WRIGHT *et al.*, 2008; LISENBEE; FORD, 2018).

The adoption of block-based programming and storytelling together is well known in the teaching and learning of programming and has proven to be effective for different student audiences. With current growth in availability of devices for augmented or virtual reality, there are possibilities for making block-based programming more immersive than before (VINCUR *et al.*, 2017; SEGURA *et al.*, 2020). In the context of SSPOT-VR, this combination is enhanced by mobile learning, which brings the characteristics of ubiquity to the teaching and learning process of programming.

In addition to the specification of the target audience, programming content, and teaching and learning method, it is also necessary to properly develop this application, considering the documentation for creating mobile VR applications. The creation of SSPOT-VR was also driven by guidelines and best practices for the development of mobile VR applications. Next, we present which documents we based the software development process of our virtual world, together with particularities from the experience of creating the application.

3.2.2 Mobile Virtual Reality

When adopting an HMD to use with SSPOT-VR, it must fulfill the requirements for applications for the Google Cardboard platform, considering that our virtual world is based on components from the Google Virtual Reality Software Development Kit (Google VR SDK)[11], used on top of the cross-platform game engine Unity 3D[12]. Google VR is the world's most accessible and affordable VR platform, supporting both Android and iOS operating systems. The SDK helps to provide immersive VR experiences merging data from devices sensors to predict the user's head position in both the real and virtual worlds[13].

Once the educational content of the application is planned, it is necessary to define the software development solutions that will be used, as well as the VR specifications that can be implemented. It is interesting to use software development solutions that are suitable for the development of VR applications, that are free to use, and have extensive documentation available. Considering the target audience of SSPOT-VR which is students from 11 years old of public and private schools, the most complete setup for using an immersive VR application in this context is characterized by the use of the mobile platform together with a low cost head-mounted display and wireless joystick.

In this context, the most suitable solution for the development of VR mobile applications is the Google Virtual Reality Software Development Kit (Google VR SDK). Google VR SDK allows to build immersive cross-platform VR experiences for Android and iOS operating sys-

[11] https://developers.google.com/vr/
[12] https://unity.com/
[13] https://developers.google.com/vr/discover/cardboard

tems. It turns a mobile device into a VR platform, enabling it to displays 3D content through stereoscopic rendering, tracks head movements in real time, and handles user interaction. It is the technology behind the Google Cardboard VR platform[14].

Google VR SDK provides support for mobile devices, head-mounted displays, and controllers. In order to be functional, it requires devices with gyroscope and accelerometer sensors that are running Android 5 or later and iOS 8 or later. Users can use VR applications with an official Google Cardboard HMD or any HMD which supports Google VR applications. The Google VR SDK provides different prefabricated artifacts that are ready to be used in mobile VR applications, such as camera controllers, event handlers, virtual objects, scripts, and even complete and functional scenes that can be modified.

However the development kit that Google VR freely provides contains all the necessary components to create virtual worlds and environments, it must be associated with a software development platform in order to be used. The SDK is compatible for native development on Android and iOS platforms, for web development, and also combined with Unity 3D Game Engine and Unreal Game Engine[15].

We chose to use the Unity 3D Game Engine since it is a platform fully equipped for creating free VR and/or AR applications when considering non-profit purposes. It enable us to create and operate interactive, real-time, and immersive experiences, providing the tools to create and publish applications across multiple platforms.

Unity enables us to create 3D scenarios by simply dropping components into the 3D scene. These scenarios contain realistic graphics and lighting effects that achieve high-fidelity VR levels without sacrificing performance. These components or assets can be found at the Unity Asset Store[16], a growing library of assets such as textures, 3D models, animations, plugins, scripts, and so on. Both Unity and community developers create assets and publish them to the store. The assets can be free or commercially affordable.

In this context, we adopted the documents provided by Google VR and Unity 3D as the basis for the design and development of SSPOT-VR: the complete documentation of Google VR SDK[8] and Unity 3D[9]. We used mainly the sections on integrating these two solutions. We have also consulted the documentation for developers of Oculus, a brand of Facebook Technologies that produces sophisticated VR devices[10]. All content presented in this Section is based on these documents and on our usage of them for the creation of SSPOT-VR.

The instructions and best practices presented in these documents are based on evaluations of early VR applications and prototypes created by the Google Cardboard design team. With fundamental aspects of human perception and cognition for VR applications, the guidelines help

[14] https://play.google.com/store/apps/details?hl=en&id=com.google.samples.apps.cardboarddemo & https://apps.apple.com/us/app/google-cardboard/id987962261
[15] https://developers.google.com/vr/develop
[16] https://assetstore.unity.com/

to design and develop VR applications for any platform.

As designing experiences in VR is different from designing traditional 2D applications, a new set of considerations for design and development is introduced, therefore understanding and following these guidelines is critical to make the application appropriate for usage. After selecting the software development solutions and guidelines with instructions for creating the application, even if they are not specific to educational applications for teaching and learning of programming, we can start to make the design decisions of the mobile VR application.

Design

The design of educational applications such as SSPOT-VR should include the usage of instructional strategies to support the learning experience. There are three strategies that the application can approach (REEVES, 2012): (i) engagement; (ii) interaction, and (iii) feedback. Users should be motivated to accomplish the tasks and enabled to properly interact within the relevant content, tasks, or challenges of the application. It is also important that all users' decisions and actions are recognized by the application which provides meaningful and instantaneous feedback to students.

The visual aspect of an application in VR is one of its differentials and must be explored in order to act positively in the engagement of its users. In order to involve users in the atmosphere of the virtual world, a theme that is known to the target audience and that fits the content of the application can be used. Sound effects can also be used to achieve similarity to the real world and to stimulate more than one sense of the target audience. Accordingly to the SSPOT-VR sci-fi theme, for all actions performed within the virtual world there is a corresponding sound that is played when the action takes place.

The SSPOT-VR virtual world lets students discover what they have to do from as few clues as possible for the purpose of fostering exploration, creativity, and aspects of learning by doing that focuses on the practice and experience of learning as a experience. The interaction with the virtual world is straightforward and based on the Google VR SDK which provides support for gaze-based interaction. It is necessary to include a form of visual cue when interacting with an object, such as the gaze-based reticle that automatically enlarge its size as a sign of a possible interaction with an object. This provides two information: first, the object is interactive, and second, the object will be selected if manual confirmation is performed.

SSPOT-VR aims to be a low cost support mechanism to the teaching and learning of programming based on mobile devices and VR. The design of the application therefore must consider low cost and accessible equipment, such as smartphones and low cost HMDs. Our application is mainly used on smartphones, allowing the usage of HMDs, although it is possible to use the application on other type of mobile devices, e.g., Tablets. What is impacted by this option is the application's immersion potential, that is reduced without the use of HMDs. For mobile VR applications created with Google VR SDK and Unity 3D, the adopted device only

needs to have the essential sensors for VR: accelerometer and gyroscope.

As we created the SSPOT-VR to be used in conjunction with an HMD compatible with Cardboard applications and wireless joystick, it is important to provide flexibility to students by providing the option of using the application as immersive or non-immersive, being the last also known as magic window mode. In both cases the same straightforward interaction must be used. From the first contact of users with the virtual world, there must be standardization in the way of communicating with them and also when providing feedback. All possible actions to be performed at that moment must be clearly informed to users, as well as the results of any action taken. Feedback must be instantaneous for everything that is modified or for any action that must be or has been performed.

Another design matter thas must be considered is the students' level of knowledge in relation to VR applications. An introductory or tutorial level is needed to present the virtual world and its functioning. Users should not be thrown immediately into intense experiences that require prompt reaction. Instead they should start experiencing the simplest interactions that ease them into the atmosphere of the virtual world. It is also recommended to always communicate important elements using words and phrases that users can understand, avoiding outside-application knowledge.

The Introduction Level must contain all the elements that users will need to know how to use in order to properly benefit from the application. This includes locomotion methods, interactions, and controls. Regardless of how the Introduction Level is made, the important aspect is that it can be understood and usable, but it is appropriate that it feels like a natural part of the virtual world. Introduction levels are commonly at the start of the virtual world as a mandatory section.

Until user do not complete the introduction, they are not able to go further. This is highly recommended as it assure that users have been given the opportunity to familiarize with the virtual world before entering the main experience. The option to experience the tutorial on demand is also a requisite. This characteristic is helpful especially for those who have cognitive disabilities or are unable to remember how to use the application.

The more time one accumulates using VR applications less likely will be to experience discomfort. The called learned comfort is a result of the brain learning to reinterpret visual irregularities that previously caused discomfort and/or vection. Vection is one of the causes of motion sickness. It is purely produced by visual stimulation and is characterized by illusions of movement of the body despite there is no movement taking place.

Incorporating resting positions and slow-paced movements into the VR experience may help minimize fatigue and motion sickness that users may feel during extended usage sessions. All of these design factors must be addressed in the creation of the virtual world. There are also instructions related specifically to the development of the mobile VR application software that

should be considered.

Development

The development of mobile VR applications using Unity and Google VR is divided between building the scenes with 3D components and programming the behavior of all scenes' components that have some interaction or behaviour. Programming the behavior of components using Unity is also called scripting and takes up most of the development of mobile VR applications.

The scenes of the virtual worlds are assembled from 3D components or assets that are purchased for use through the Unity Asset Store itself[16]. A Unity asset is an component that is ready to be used or modified. An asset may come from a file created outside of Unity, such as a 3D model, an audio file, an image, or any of the other types of file that Unity supports.

While it is also possible to create their own 3D assets, the Asset Store offers a vast catalog of free and paid assets that can be useful for developers. It is possible to combine assets from various sources in order to create the virtual world as stipulated in its design. Design decisions however may have to be modified according to what is available in the asset store.

As a result of building up the scenes, with all possible interactions and planned actions, it is necessary to create the scripts that perform these actions, as well as associate them with their respective components. It is the scripts that give life to the virtual world. Unity receives the instructions from the scripts and executes them frame after frame as fast as it can. Achieving a high frame rate means not only the graphics will look fluid, but controls will be more responsive. The programming language that is used in Unity 3D is C#. The programming method proposed in SSPOT-VR is based on block-based programming principles and was developed with simple 3D components, such as cubes, and scripting, including the scripting of the compiler that processes the algorithm created by the student. The student's movement is also based on scripting, as well as the movement and behavior of all components of the virtual world.

As VR allows users to take in information and content in the same visually way as in the world, it has the potential to present mismatches between physical and visual motion cues. This mismatch can produce nausea known as motion sickness. Optimizing performance is therefore critical for creating a motion sickness free user experience. It is necessary to keep the virtual world rendering at 60 frames per second to maintain VR immersiveness and prevent any discomfort. Besides that, the camera's movement should always represent the user's view and respond to the user's motion, regardless of which camera or viewpoint is used.

VR is computationally expensive in comparison to applications that do not use VR, primarily due to having to render the stereoscopic view, i.e., everything once per-eye. In addition to that, mobile VR can be particularly demanding. It is important to take mobile platform limitations into account when designing mobile VR experiences.

Another important factor is the motion-to-photon latency, that is the point in time when a motion occurs and the point in time when the resulting image becomes visible to the user. Minimizing motion-to-photon latency is the key to giving the user the impression of presence. It is recommended to target 20 milliseconds or less motion-to-photon latency for any input made via VR hardware to be reflected on the device's display.

In addition to that, it is generally suggested that developers of mobile VR applications: (i) adopt non-directional lights for static 3D assets and light probes for dynamic 3D asssets; (ii) minimize or eliminate the usage of post-process effects; (iii) keep geometry as simple as possible and avoid using computationally heavy shaders; (iv) minimize draw calls, and (v) use Physics wisely as it is computationally heavy.

As presented, we based the planning and development of SSPOT-VR on teaching and learning of programming guidelines and on the design and development documentation for mobile VR applications. We established an application foundation that can be expanded, in order to contain new challenges that address new programming concepts. Next, we present the design for two new levels of SSPOT-VR.

3.3 Final Remarks

This chapter presented SSPOT-VR, a Space Station for Programming Trainning in Virtual Reality. We presented how the mobile immersive VR block-based programming application enables children and teenagers to learn and practice programming in a virtual space station. The main idea of the application is to support primary and secondary students in the process of learning and practicing the concept of algorithm. We described each level of SSPOT-VR, as well as the story we created to engage students in the application, and the programming method we adapted using block-based programming principles for assisting the programming practice and user interaction.

SSPOT-VR, as a complete and functional application, collaborates with the idea that its development is feasible and that it is possible to present different programming contents in an interactive and immersive or non-immersive way through a mobile VR application. We assessed what works and what does not fit to this context accordingly to the teaching of programming and mobile VR software development guidelines we used. We also provide the documents we used and add our experiences to support the construction of applications similar to SSPOT-VR.

SSPOT-VR differs from other VR applications that support the teaching and learning of programming, particularly, because it is based on a mobile platform. Another difference is that SSPOT-VR is compatible with the use of HMDs, to provide greater immersion to its users. In addition to that, it differs since its construction is based on organized and complete documents to obtain an objective: the teaching and learning of programming for children and teenagers. These documents are the SBC (2019) programming teaching guidelines and the reference curriculum

of Raabe, Brackmann and Campos (2018), that provides thorough plannings for the teaching of Computing, and not just programming, for all years of primary and secondary education.

The main contribution of this Master's work is SSPOT-VR. To the best of our knowledge, there was no mobile application that uses immersive VR, mobile devices, block-based programming, and storytelling to support the teaching and learning of programming to children and teenagers.

Next, we present two studies conducted for evaluation of the SSPOT-VR application.

CHAPTER

4

SSPOT–VR EVALUATION

The conduction of evaluations enable us to measure the achievement of pre-established objectives of the SSPOT-VR application, i.e., identify whether or not the mobile VR application is able to accomplish individual acceptance by a specific group of users; by acceptance we mean adequate levels of intuitiveness, interaction, and immersion. Evaluations are an essential stage for the development any software application. They support the identification of strengths and weaknesses in the relationship between users and the applications. Particularly in the case of a VR application, the user experience can be evaluated using criteria such as: usability, learning, fun, immersion, presence, comfort, satisfaction, emotion, attractiveness, and perception (TORI; HOUNSELL, 2018).

In this chapter, we present two evaluations conducted in the scope of SSPOT-VR. The first is an evaluation with the target audience. Even though it is recommended by SBC (2019) and Raabe, Brackmann and Campos (2018) that children and teenagers begin to use electronic devices to learn at the age of 11, every student of primary and secondary schools can potentially use SSPOT-VR. This evaluation regards the user acceptance and use of technology. The second evaluation is an usability evaluation conducted by experts.

The objective of these evaluations include verify whether students would adopt SSPOT-VR for usage as well as identify usability improvements to be applied in the future for the expansion of the SSPOT-VR application. We chose to evaluate the user acceptance and usability before assessing the learning potential as we need to assess the feasibility of building and using applications as SSPOT-VR for use in educational environments. Another motivation for this is that, to the best of our knowledge, SSPOT-VR is the only mobile immersive VR application that uses block-based programming and storytelling to support programming learning for children and teenagers.

In Section 4.1, we present an evaluation of SSPOT-VR conducted with its target audience to measure the acceptance, usefulness, and enjoyment levels of the application. We want to

understand the extent to which the user believes that using a mobile VR application is adequate to learn about programming. To collect data from this quantitative survey research, we used questionnaires with questions adapted from three similar studies: (i) a study that proposes techniques to measure the acceptance and use of general technology (VENKATESH *et al.*, 2003); (ii) a study that presents a scale to measure the enjoyment of students when using educational games (FU; SU; YU, 2009), (iii) and a study that presents a path model to examine students' perceptions when using educational games (BOURGONJON *et al.*, 2010).

In Section 4.2, we present an evaluation carried out to measure the usability levels of SSPOT-VR regarding the intuitiveness of the interaction and sense of immersion provided by the application. This inspection-based evaluation method was performed by experts and is based on twelve specific usability heuristics to evaluate VR applications (SUTCLIFFE; GAULT, 2004). The specialists firstly performed a technology audit of SSPOT-VR to demonstrate their impressions about the application according to some parameters. After that, they listed usability issues of SSPOT-VR along with their severity as they performed a list of pre-defined tasks.

4.1 User Acceptance and Use of Technology

Studies conducted with the target audience are generally associated with evaluating the application requirements, i.e., verifying that users' needs are met. In the context of VR applications, requirements associated with visualization, interaction, and immersion are among the most important requirements for the user. When it comes to educational and training applications, these requirements take on even more importance, since applications with this purpose can be more effective if a high degree of realism is achieved (TORI; HOUNSELL, 2018).

Taking into consideration that SSPOT-VR may not be common application to students and that it was built to fill a gap in the research area of teaching and learning of programming, we evaluated it with the target audience considering their perspective of acceptance, usefulness, and enjoyment associated with the application. Therefore, evaluating aspects of the application that influence the acceptance, usefulness, and enjoyment help us to understand the extent which the user believes that adopting a mobile VR application is adequate to learn programming concepts and practices.

Works similar to SSPOT-VR conducted the evaluation using different methods. Vincur *et al.* (2017) and Segura *et al.* (2020) presented an immersive virtual worlds based on a sophisticated VR setup and used questionnaires with children, teenagers, and adults to collect their impressions associated with the applications. Their questionnaires contain questions related to appreciation, preference, difficulty level, and learning opportunity. The questions were designed to be answered using a Likert scale. Results were presented based on descriptive statistics, such as the average and standard deviation of the score for each question.

Tanielu *et al.* (2019) presented an immersive virtual world to be used with a mobile

head-mounted display, although not specifying the exactly adopted VR setup. They also relied on questions with answers in the form of a Likert scale to evaluate their application. They asked undergraduate students how strongly they prefer physical and hands-on activities to learn on comparison with visual learning, that requires to imagine things, or textual learning, that requires reading. In addition to the descriptive statistics, they also provided correlation analyzes of their questionnaire responses, applied in pre-test and post-test format. Another common aspect of these works is that they are not based on previously evaluated studies that propose questions to compose a questionnaire to evaluate their VR application. The questionnaire we used to evaluate SSPOT-VR is composed by questions that were presented in evaluated studies.

There are no evaluation methods endorsed exclusively to analyze mobile VR applications that support the teaching and learning of programming to children and teenagers. Since our goal was to evaluate the acceptance, usefulness, and enjoyment of SSPOT-VR. To achieve our goal, we tailored into a questionnaire the Unified Theory of Acceptance and Use Of Technology (UTAUT) presented by Venkatesh *et al.* (2003); the EGameFlow scale, presented by (FU; SU; YU, 2009) to measure learners' enjoyment of e-learning games, and the path model proposed by Bourgonjon *et al.* (2010) to examine the perceptions of students when using educational games.

UTAUT aims to explain users' intentions to use an application that they have never seen before, as well as their subsequent usage of the application. The unified theory combines eight studies that are associated with the acceptance and use of technology. The union of these studies resulted in four key categories of items that measure the intention of technology usage (VENKATESH *et al.*, 2003): (i) performance expectancy; (ii) effort expectancy; (iii) social influence, and (iv) facilitating conditions. The fourth is a direct determinant of user behavior while the others are determinants of usage intention and behavior.

Among the studies we tailored, the UTAUT model to their usage contexts and quantitatively evaluate the collected data, the study of Leal *et al.* (2011) aims to identify factors that determine the acceptance and usage of ICTs in distance education, in the context of the instructors. The work of Silva and Rodello (2017) presents an acceptance evaluation with potential customers of an AR web application that function as marketing strategy. Finally, the work of Almaiah, Alamri and Al-Rahmi (2019) analyzes the acceptance of mobile learning applications with undegraduate and graduate students.

In addition to the items that UTAUT contemplates, we identified the need for more specific questions to evaluate our mobile VR application. We therefore used as a basis other two studies to compose the questions in our questionnaire. Although it is not possible to say that they are the same, games can be considered very similar to VR applications, mainly due to their realistic graphics. The engagement in a VR application, a game's entertainment value, and other similar sensory experiences also lead to the users' involvement in the learning activity. Mostly factors associated with users' motivation to learn are derived from interaction processes (FU; SU; YU, 2009).

Despite being created to measure educational games, the EGameFlow scale has questions that are consistent with the context of SSPOT-VR. The scale contains different questions associated with the categories of concentration, goal clarity, feedback, challenge, autonomy, immersion, social interaction, and knowledge improvement (FU; SU; YU, 2009). Among these categories and considering SSPOT-VR, the categories of concentration, goal clarity, feedback, immersion, and knowledge improvement stand out.

We also used the path model for understanding students' acceptance of educational games proposed by Bourgonjon *et al.* (2010). The scale presented in this study is based on the Technology Acceptance Model (TAM), also covered by the UTAUT, and has items according to the categories usefulness, ease of use, learning opportunities, experience with games, and preference for games. In our case, we highlight usefulness, learning opportunities, and preference for VR applications.

4.1.1 Goals

This quantitative exploratory survey research main goal is to answer the research question of this Master's work. We want to understand if students of primary and secondary public and private schools would accept the usage of a mobile immersive VR application to support the learning of programming. To answer it, we need to identify the extent which students believe that using a mobile VR application is adequate to use to learn about programming, in addition to verifying the levels of appreciation of students in using the application in the classroom.

Therefore, **we aim** to analyze SSPOT-VR **for the purpose** of evaluation, **with respect** to its acceptance and use, **from the viewpoint** of children and teenagers from primary and secondary public and private schools, **in the context** of the classroom.

4.1.2 Subjects

We conducted online and free programming workshops in order to evaluate SSPOT-VR. A total of 124 students of primary and secondary public and private schools voluntarily participated in the evaluation of SSPOT-VR.

The educational stages in Brazil are divided between what we call elementary school (*ensino fundamental*) and high school (*ensino médio*), as Table 1 reveals. These two stages include the primary and secondary education model. The elementary school starts when students are usually 6 and lasts for 9 years, while high school starts when students are usually 15 and lasts for 3 years.

Students from almost all grades from elementary school and high school participate in the workshop and experienced SSPOT-VR. Only students from the school grades elementary school 1 and 2 did not participate in any of the programming workshops. A 6 year old student also participated in one of the workshops. At this age the student has not started elementary

Table 1 – School grade

Students' school grade	**N**	**%**
Preschool	1	0,81
Elementary school 3	4	3,23
Elementary school 4	8	6,45
Elementary school 5	5	4,03
Elementary school 6	24	19,35
Elementary school 7	22	17,74
Elementary school 8	12	9,68
Elementary school 9	10	8,06
High school 1	16	12,90
High school 2	10	8,06
High school 3	12	9,68
Total	**124**	**100,00**

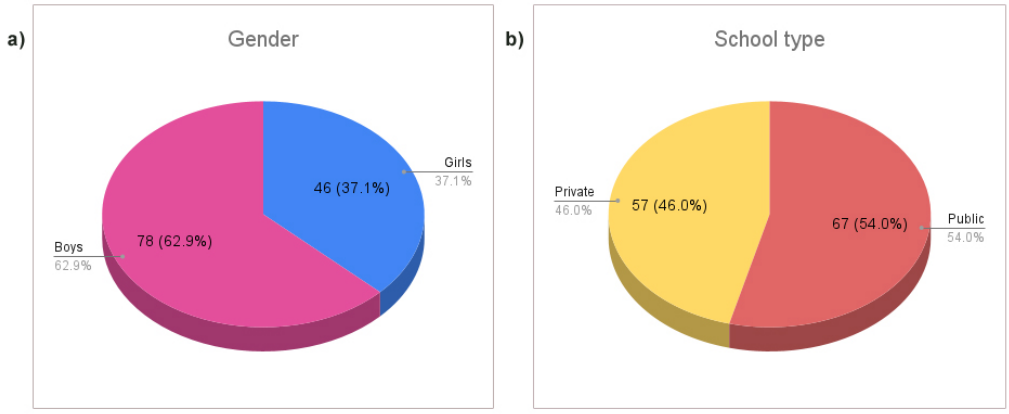
school yet, being part of an educational stage called preschool (*pré-escola*). The variance in the students' grades who participated in the workshops is consequently associated with a high range of ages, as Table 2 reveals. This variation was allowed for the reason that we chose to evaluate whether SSPOT-VR can be used for a wide range of students in a real teaching and learning environment. As presented in Section 4.1.4, although the age variation, students' impressions of the SSPOT-VR application were similar.

Table 2 – Students' ages

Age	**N**	**%**
6	1	0,81
8	2	1,61
9	5	4,03
10	8	6,45
11	15	12,10
12	20	16,13
13	17	13,71
14	16	12,90
15	10	8,06
16	15	12,10
17	14	11,29
19	1	0,81
Total	**124**	**100,00**

Analyzing both tables, it is possible to verify that the sample is composed mostly of students aged 12 and 13, enrolled in elementary school 6 or 7. As Figure 46 illustrates, the workshops that we conducted had a greater presence of boys, which represented 62,9% of the subjects, while girls are 37,1%. Regarding the school type, which can be public or private in our context, our sample was balanced.

Figure 46 – a) Subjects gender b) Subjects school type



Source: Author

4.1.3 Method

Due to COVID-19, we were unable to conduct the evaluation within the classroom. So, we conducted a set of online and free programming workshops for elementary and high school students. The workshop took place in four different dates, nevertheless each one comprised exactly the same content.

To conduct and advertise the workshops, we had support from GRACE (*Grupo de Alunas nas Ciências Exatas*) of *Instituto de Ciências Matemáticas e de Computação* (ICMC) from the *Universidade de São Paulo* (USP), the school board committee of *São Carlos*, and local newspapers.[1,2,3]

SSPOT-VR was created to be immersively experiencied through a VR adapter, viewer, or head-mounted display that satisfies the Google Cardboard standard. Due to the fact that the workshop took place online, SSPOT-VR was experienced by students directly in the mobile device display without stereoscopy or the usage of a VR adapter, offering standard immersion. In both scenarios, the requirement to use the application is the same. It is necessary to use a mobile

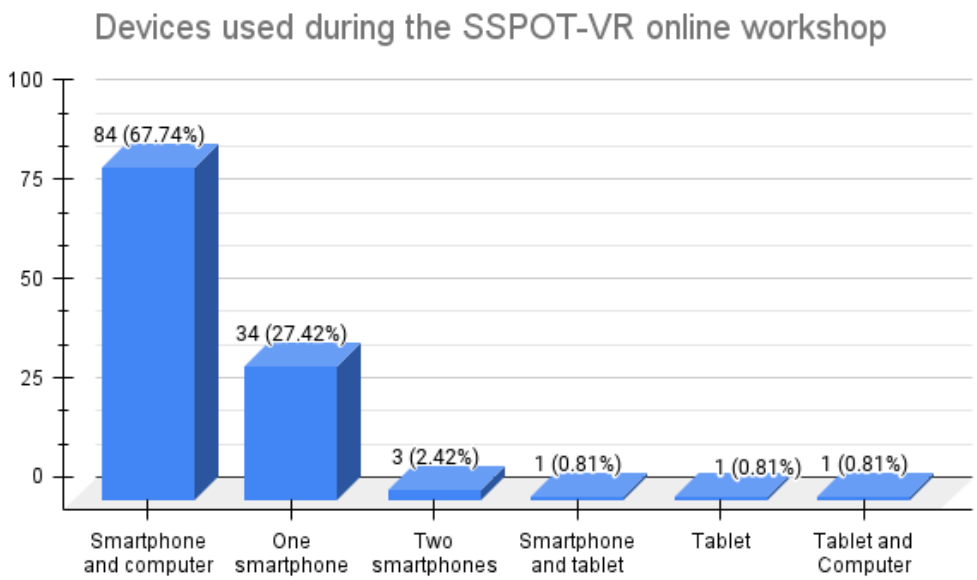
[1] https://www.saocarlosagora.com.br/cidade/oficina-online-na-usp-sao-carlos-aprenda-principios-de-programacao/131495/
[2] https://jornal.usp.br/universidade/oficina-on-line-ensina-nocoes-de-programacao-dentro-de-uma-estacao-espacial/
[3] https://globoplay.globo.com/v/9049258/programa/

device (preferably a smartphone) that is compatible with Google Cardboard, i.e., the device must contain an accelerometer and gyroscope sensors.

Before students start to effectively use SSPOT-VR, we briefly presented the application context and provided basic usage instructions. Next, all students simultaneously performed interactions with SSPOT-VR, being guided by the researchers when needed. The session duration was approximately 2 hours for a class of up to 30 students.

Students experienced SSPOT-VR and participated in the online workshop accordingly to a combination of device usage, usually smartphone and computer. Even though in some cases students also used the same smartphone to experience SSPOT-VR and attend the workshop. Figure 47 reveals the devices students used to participate in the workshop and experience SSPOT-VR.

Figure 47 – Combinations of devices students adopted to attend the workshop



Source: Author

The user acceptance evaluation of SPPOT-VR is characterized by the usage of the application by the target audience followed by the application of an online questionnaire with 20 questions based on the UTAUT model (VENKATESH *et al.*, 2003), on the EGameFlow scale (FU; SU; YU, 2009), and on the path model of (BOURGONJON *et al.*, 2010). The answers for the questions are based on a 5-point Likert scale, with 1 representing "strongly disagree" and 5 "strongly agree".

It should be noted that the questionnaires were previously tested on a sample of 4 volunteers in order to identify possible questions and problems with the formatting of the questionnaire. It is also important to mention that we followed the guidelines from the *Comitê de Ética em Pesquisa (CEP)* to conduct the workshops and collect data. The guidelines provide

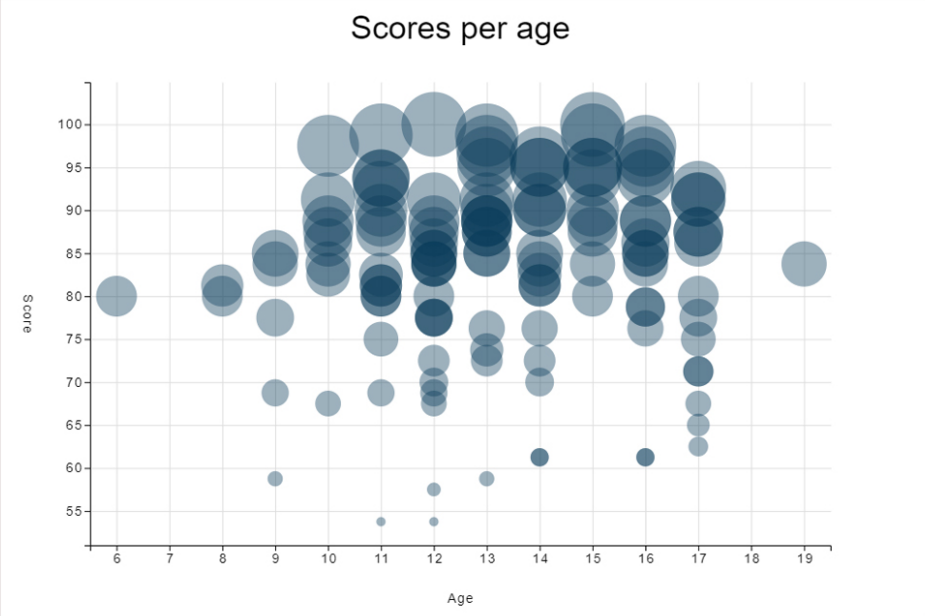
regulatory standards for research involving humans. The purpose for its creation is to defend the interests of the research subjects in their integrity and dignity and to contribute to the development of research within ethical standards[4]. In order to participate in the workshops, students and their guardians formally expressed their agreement to be part of this research.

Considering that we adopted a survey research method to collect descriptive quantitative data, we present and analyze the data by using descriptive statistics and exploratory factor analysis (HAIR *et al.*, 2010). The factor analysis, along with other statistics tests, such as the Kaiser-Meyer-Olkin (KMO), and the Cronbach's Alpha, were performed using the SPSS Statistics software[5].

4.1.4 Results

In order to perform a descriptive analysis of the collected data, we calculate the questionnaire score for each student, as well as for each question. The objective of reducing the questions to a score is to be able to measure the students' perceptions about SSPOT-VR in a scale from 0 to 100, whereas 100 is the optimal value. We assembled the 20 questions from our questionnaire into scores and distributed these scores to students according to their age, as the scatter plot in Figure 48 reveals.

Figure 48 – Scatter plot



Source: Author

Although it is possible to notice scores with lower values, it is also possible to verify that the concentration of the students' scores remained above 80 during almost the entire age

[4] https://bvsms.saude.gov.br/bvs/saudelegis/cns/2013/res0466_12_12_2012.html
[5] https://www.ibm.com/analytics/spss-statistics-software

range of the participants of our workshops. The average for the students' scores is 82.785 while the standard deviation is 10.788. There are 51 of 124 scores lower than the average, while there are 72 scores higher than the average, i.e., more than 58% of the sample of 124 students scored equal or higher than the average of 82,785.

Among boys, 35 of 78 boys' scores were below the average score, while 42 were above it, this represents that more than 53% of boys' scores were above the average score. Regarding girls, 16 of 46 girls' scores were below the average score, while 30 were above it, this represents that more than 65% of girls' scores were above the average.

Concerning the school type, that could be private or public, the same amount of students' scores were above the average. Among 57 students from private schools, 36 scored above the average, representing more than 64% of private school students. For public schools, more than 53% scored above the average, i.e., 36 of 67 students' scores from public schools were above the average.

A different perspective is presented in Chart 5, that reveals all of the questions that our questionnaire contains, together with its source and the average score and standard deviation for each question.

This chart reveals that most of the positive questions, i.e., questions that the answer should be closer to 5 (strongly agree), obtained an average score above 4. While the negative questions (questions 2, 4, and 9) did not reached an average score of 2, considering in this case that the closer to 1 (strongly disagree), the better. However, some positive questions had their averages below 4, as is the case of questions 6 and 7, which also have a higher number for standard deviation. The same is true for questions 15 and 19, with an average below 4 and high variation. Although they do not score far from the ideal, high variations in the answers also influence questions 4 and 9. On the other hand, high averages and low variation stand out in questions 10 and 11.

To better present the results of these highlighted questions, we present the pie charts of each one of them, revealing the frequency of responses for each one of the Likert scale options. Questions 6 and 7 present high variations between answers, as can be seen in Figure 49. The objective of these questions is to measure students' perceptions regarding the immersion provided by SSPOT-VR. It is notable that part of the students disagree that they forget about time and the surroundings while using the application, as is also notable the students who were in doubt and answered that neither agree nor disagree. As explained in Subsection 4.1.3, during the workshops, students used SSPOT-VR directly on the mobile device, compromising immersion.

Despite the fact that the results for questions 15 and 19 are farther from the maximum than the other questions, they are not representing low scores, as Figure 50 reveals. Even so, there was a wide variation in answers and a high frequency of answers for neither agree nor disagree. Students response, or lack of a more meaningful response, reveals that the students,

Chart 5 – The complete questionnaire with average and standard deviation for each question

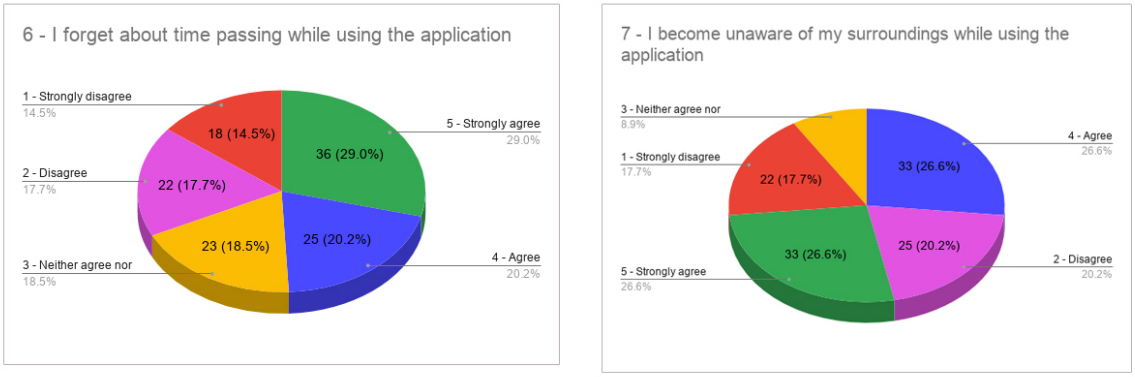
ID	Source	Statement	Average	S. D.
1	Fu, Su and Yu (2009)	I can remain concentrated in the application.	4,67	0,70
2	Fu, Su and Yu (2009)	I am burdened with tasks that seem unrelated to learning programming.	1,41	0,71
3	Fu, Su and Yu (2009)	Overall application's goals were presented clearly in the beginning.	4,73	0,70
4	Fu, Su and Yu (2009)	Workload in the application is too much.	1,82	1,29
5	Fu, Su and Yu (2009)	I receive immediate feedback on my actions.	4,35	1,10
6	Fu, Su and Yu (2009)	I forget about time passing while using the application.	3,30	1,43
7	Fu, Su and Yu (2009)	I become unaware of my surroundings while using the application.	3,23	1,48
8	Venkatesh *et al.* (2003)	I could complete a job or task using the application if there was no one around to tell me what to do as I go.	4,27	1,23
9	Venkatesh *et al.* (2003)	The application is somewhat intimidating to me, I hesitate to use it for fear of making mistakes.	1,93	1,28
10	Fu, Su and Yu (2009)	I understood the basic idea of algorithms.	4,74	0,63
11	Fu, Su and Yu (2009)	I want to know more about algorithms and programming.	4,76	0,53
12	Bourgonjon *et al.* (2010)	Applications like this one offer opportunities to learn new things in a cool way.	4,76	0,51
13	Venkatesh *et al.* (2003)	I like working with the application.	4,78	0,52
14	Venkatesh *et al.* (2003)	I have the resources necessary to use the application.	4,20	1,12
15	Bourgonjon *et al.* (2010)	Using applications like this one in the classroom would help me to achieve better grades.	3,98	1,00
16	Venkatesh *et al.* (2003)	A specific person or group is available for assistance in case I have difficulties with the application.	4,54	0,75
17	Bourgonjon *et al.* (2010)	I would know how to handle applications like this one in the classroom.	4,54	0,76
18	Bourgonjon *et al.* (2010)	It would be easy for me to use applications like this one in the classroom.	4,30	0,98
19	Venkatesh *et al.* (2003)	My family and professors think that I should use applications like this one.	3,67	1,04
20	Bourgonjon *et al.* (2010)	If I had to vote, I would vote in favor of using applications like this one in the classroom.	4,56	0,89

their instructors, and family are unlikely to be aware of applications similar to SSPOT-VR in order to express their opinion about usage or usage influence.

Although questions 4 and 9 resulted in a standard deviation above the other questions, it is possible to observe in Figure 51 that most of the answers were still for the strongly disagree (1) and disagree (2) options. These results, therefore, show that most students consider that SSPOT-VR has an adequate workload and that there is no difficulty or fear involved in its use.

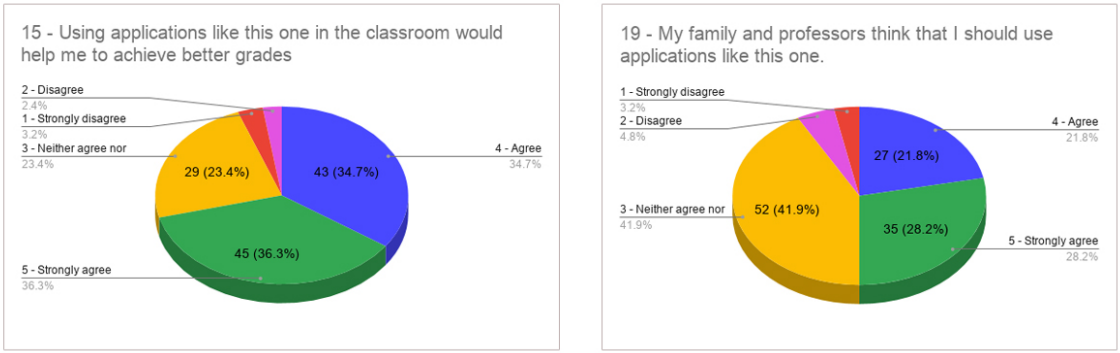
The lowest rates of variation and highest averages occur with questions 10 and 11, which measure students' perception of the learning opportunities created by using SSPOT-VR. As illustrated in Figure 52, the results show that students understood the basic idea of algorithm and that they would like to learn more about this topic.

Figure 49 – Pie charts for the questions 6 and 7



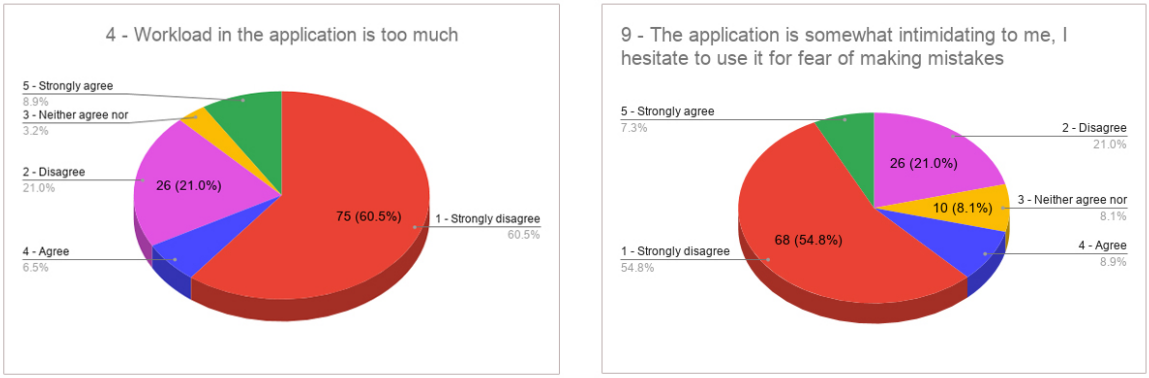
Source: Author

Figure 50 – Pie charts for the questions 15 and 19



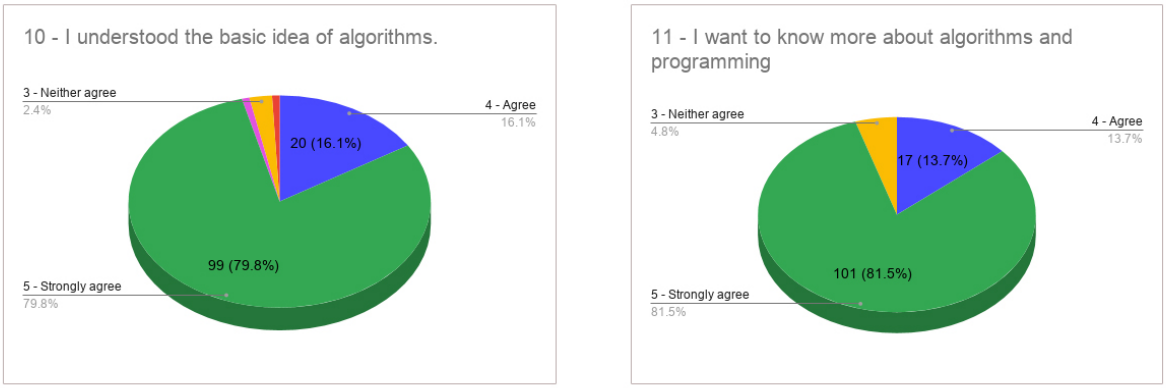
Source: Author

Figure 51 – Pie charts for the questions 4 and 9



Source: Author

Figure 52 – Pie charts for the questions 10 and 11



Source: Author

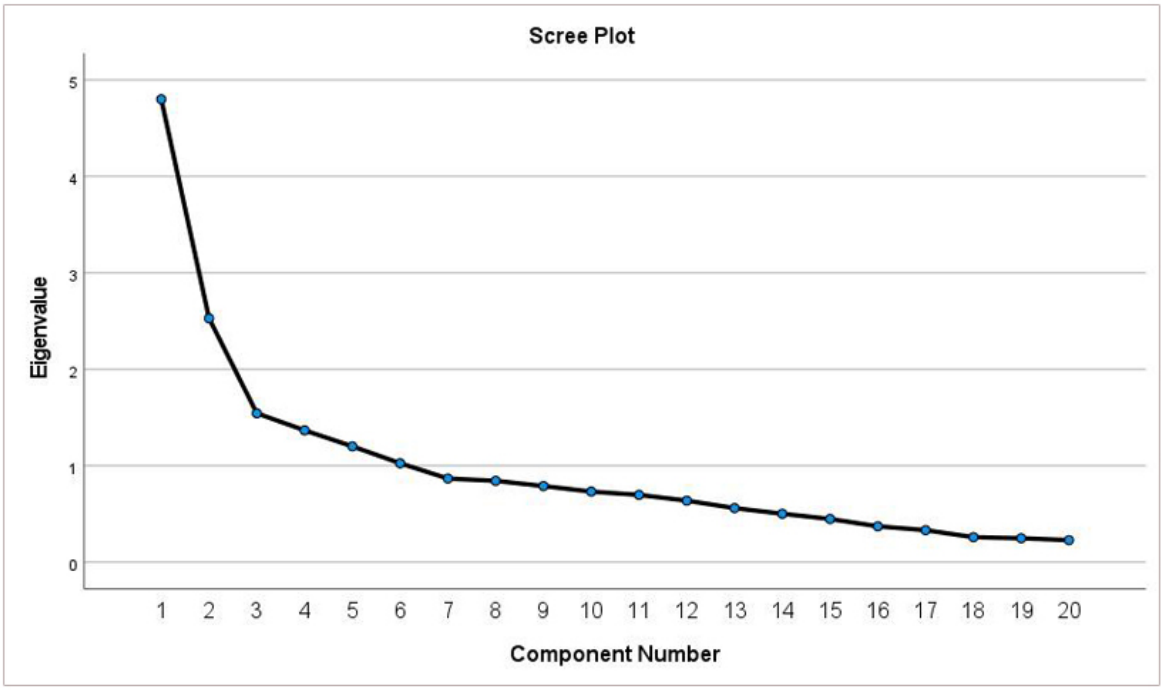
In order to complement the descriptive statistics, we conducted an exploratory factorial analysis with the collected data. Exploratory factor analysis is a set of multivariate techniques that aim to find the underlying structure in a data matrix to determine the number and nature of factors that best represent the set of all variables. Analyzing the structure of the interrelationships of a given number of variables makes it is possible to define the factors that best explain their covariance. Variables belong to the same factor when they share a common variance (HAIR *et al.*, 2010).

When applied to our sample, the Kaiser-Meyer-Olkin (KMO) test had a result equal to 0,764, indicating that factor analysis is adequate for our data. The KMO test is a measure of how suited the data is for factor analysis, it measures sampling adequacy for each variable in the model and for the complete model (HAIR *et al.*, 2010). According to Hutcheson and Sofroniou (1999), as a rule for interpreting KMO indices, values between 0,7 and 0,8 are considered satisfactory to measure the adequacy of data for exploratory factor analysis.

The scree plot can help to identify the number of strong factors and their eigenvalues, which represent the total variance divided by each one of the factors. The plot presents the variance from the largest to the smallest value by each strong factor. The principle of the scree plot is to retain only those components that are above the point after which the remaining eigenvalues linearly decline (HAIR *et al.*, 2010). It is possible to verify in Figure 53 that the eigenvalues of our data decline approximately linearly after the sixth component. We decided, for this study, to retain the six first eigenvalues.

There was a reduction of a problem with 20 factors to 6 strong factors, as Table 3 illustrates. In order to obtain the factors, the extraction method used was the Principal Component Analysis and the rotation method was the Varimax with Kaiser Normalization. The composition of the factors was based on the selection of questions with loads greater than or equal to 0,40

Figure 53 – Scree plot



Source: Author

(HAIR *et al.*, 2010). All statistics were performed by SPSS Statistics.

Analyzing Table 3 and considering that the total of variables in the original set is 20, it is possible to verify that only the first eigenvalue explains 4,801/20 = 24.003% of the variation. The second highest eigenvalue is 2,529, which explains 12,647% of the variance of the original data, followed by 7.723% of the third factor, 6,839% of the fourth factor, 6,010% of the fifth factor, and 5,132% of the sixth factor. Together, the 6 strong factors assume a cumulative variance of 62.354%, revealing that more than half of the variance in the data is associated with these factors.

Lastly, the Cronbach's Alpha was used to verify the scale reliability or internal consistency levels of our questionnaire (HAIR *et al.*, 2010). The Cronbach's Alpha measures the correlation between responses on a questionnaire by analyzing the average correlation of responses, it provides how closely related the set of items is as a group. The result for our data was Cronbach's Alpha = 0,691, demonstrating the reliability of the questionnaire. According to Landis and Koch (1977), a value for Alpha greater than 0,6 is considered good for assessing the reliability of the questionnaires. It is important to note that the removal of any question did not significantly influenced the value of the Cronbach's Alpha, either for an improvement or a worsening, in relation to its optimal parameters.

Table 3 – Rotated and normalized component matrix

	Questions	Factors					
		F1	**F2**	**F3**	**F4**	**F5**	**F6**
13	I like working with the application	,758					
8	I could complete the task using the application if there was no one around to tell me what to do as I go	,729					
1	I can remain concentrated in the application	,724					
10	I understood the basic idea of algorithms	,597					
12	Applications like this one offer opportunities to learn new things in a cool way	,556					
11	I want to know more about algorithms and programming	,514					
20	If I had to vote, I would vote in favor of using applications like this one in the classroom		,888				
18	It would be easy for me to use applications like this one in the classroom		,835				
16	A specific person or group is available for assistance in case I have difficulties with the application			,688			
15	Using applications like this one in the classroom would help me to achieve better grades			,628			
19	My family and professors think that I should use applications like this one			,583			
17	I would know how to handle applications like this one in the classroom			,530			
2	I am burdened with tasks that seem unrelated to learning programming			-,505			
14	I have the resources necessary to use the application				,794		
9	The application is somewhat intimidating to me, I hesitate to use it for fear of making mistakes				-,713		
4	Workload in the application is too much				-,448		
7	I become unaware of my surroundings while using the application					,838	
6	I forget about time passing while using the application					,796	
3	Overall application's goals were presented clearly in the beginning						,805
5	I receive immediate feedback on my actions						,550
	Eigenvalue	4,801	2,529	1,545	1,368	1,202	1,026
	% Variance	24,003	12,647	7,723	6,839	6,010	5,132
	% Cumulative Variance	24,003	36,650	44,373	51,212	57,222	62,354

4.1.5 Discussion

Analyzing the average scores for each student and each question, it is possible to identify in both cases a trend towards optimal values. This trend indicates the favorable perceptions of primary and secondary school students about the use of SSPOT-VR. The predominance of scores close to the optimal values reveals that a mobile learning application which uses virtual reality, block-based programming, and storytelling to support the learning of programming for children and teenagers, is suitable for use by such audience who is part of a wide-range of ages and grades across the entire compulsory educational stage.

The specific SSPOT-VR target audience represents approximately 37% of the sample, i.e., students who may be in the elementary school 6 or 7. Students younger than the target audience of SSPOT-VR, that are below elementary school 6, represent approximately 15% of the total sample. Older students are also interested in SSPOT-VR. Approximately 48% of students who participated in the workshops are above elementary school 7. Among the 48%, 30% are students who are no longer in elementary school and are already attending high school.

In addition to the high average scores for each question and each student, the exploratory factorial analysis of the data we collected with the questionnaire generated 6 strong factors. Each factor contains questions from 2 or more different studies between the studies of Fu, Su and Yu (2009), Bourgonjon *et al.* (2010), and Venkatesh *et al.* (2003), which we used as basis to create our own questionnaire.

Factor 1 is our strongest factor in terms of variance and defines that the user acceptance and use of SSPOT-VR depends on the learning opportunities that the application provides, as well as the affection students feel when using the application. Factor 1 also relates to self efficacy, i.e., ability to use and perform actions autonomously within the application. Students naturally reveal affection and preference for learning tools that address their mindsets through the manipulation of learning objects that allow interaction, critical thinking, control, and experimentation. The concept of learning opportunities is theoretically related to usefulness, then learning opportunities will positively affect usefulness as well. Therefore, perceived learning opportunities have an important role on the acceptance of educational applications (BOURGONJON *et al.*, 2010).

Factor 2 influences the acceptance and use of SSPOT-VR in relation to its degree of ease of use. Ease of use refers to the degree to which a person believes that using a particular system would be free of effort. The maximum goal is to provide an experience that provides deep and effortless involvement. People will usually perceive a certain type of technology as more useful when it is easy to use (BOURGONJON *et al.*, 2010).

Factor 3 is related to usefulness and social influence. It helps to investigate whether students would consider using SSPOT-VR in the classroom and whether this use would help them to achieve better grades. It also has to do with the degree to which students believe that using SSPOT-VR would enhance her school performance, as well as whether there are people in the students' circle who believe that they should use the application, such as parents and instructors. Extrinsic motivation is operationalized using the same items as perceived usefulness, i.e., doing something for the purpose of obtaining a reward (VENKATESH *et al.*, 2003).

Factor 4 relates to aspects of facilitating conditions. Unlike ease of use, facilitating conditions refers to the resources and level of difficulty involved in using the application. For instance, the presence of objective elements in the environment that users agree that it makes an act easy to accomplish. It is also important to note if the application causes any negative feelings in its users while it is being used, as evoking anxious or emotional reactions (VENKATESH *et al.*, 2003).

Factor 5 is about immersion. The immersion potential of SSPOT-VR was burdened due to the usage setup we had to resort in order to conduct the online workshops. Even so, we checked whether students reveal signs of immersion, such as forget about time passing while using the application or become unaware of surroundings while using the application. Usage setup affects students' sense of immersion (TORI; HOUNSELL, 2018). Even though VR applications with different levels of entertainment, feedback, mission, sense of triumph, and social interaction, still support students' immersion sense (FU; SU; YU, 2009).

Factor 6 relates to aspects of goal clarity and feedback, accordingly to how efficiently the application communicates with users and whether the objectives of the application are clear. It is essential to provide clear information on how the participants are doing and what is possible for the user to take action within the application at any given time. All this information must be clearly communicated to students at all times. Feedback must be given regarding the learner's response (FU; SU; YU, 2009).

Analyzing the descriptive statistics and the exploratory factor analysis, the results indicate that students accept the use of SSPOT-VR and see benefits associated with its usage. Actually, the scores per student were high overall and the scores per question also were high overall, even though some questions showed high variation and scored below the high average. Furthermore, the better the learning opportunity, or how it is perceived by students, and how the affection of students will subjectively act, the greater the chance that SSPOT-VR will be accepted by students for use. Affection involves the natural interest in using the application, the application must capture the student's attention and be usable without any limitations or problems.

We state that students accept to use SSPOT-VR, appreciate its use, and believe they can learn something using it, as we obtained a large sample to test SSPOT-VR. While we evaluated SSPOT-VR with its specific target audience, elementary school 6 students aged approximately 11, we were also able to evaluate SSPOT-VR with students of nearly every educational stage, with ages from 6 to 19 years old. With this, we obtained broad impressions about SSPOT-VR, although most were positive in relation to usefulness, learning potential, ease of use, entertainment, and affection provided.

To the best of our knowledge, SSPOT-VR is the only mobile VR application that uses block-based programming and storytelling to support the learning of programming for children and teenagers. This is the main reason we evaluated its acceptance, usefulness, and enjoyment. Before evaluating the learning potential it provides, we needed to make sure there is feasibility in planning, building, and releasing for wide use an application such as SSPOT-VR. These results should be taken into account when instructors are considering using the application in the classroom, either online or in person, with or without head-mounted displays, in computer science, robotics, or cross-curricular in non-programming or computing disciplines. Although it is important to consider the generalizability of the results discussed here, it is important to observe that they were limited by the context in which SSPOT-VR was used during the

workshops, which took place online, with the monitoring of researchers, and without the use of head-mounted display .

Next, we present threats to validity of the evaluation presented along this section.

4.1.6 Threats to Validity

An experimental design study, e.g., a two-group control design was not performed. Before comparing SSPOT-VR with another tool to support the learning of programming to children and teenagers or evaluating student learning level using SSPOT-VR, we must evaluate the application user acceptance. We choose to take this evaluations to recognize whether students are interested in using SSPOT-VR and whether they would adopt it for usage. In this context, we focused on the achievement, in decreasing order, of the following validities: internal, external, construct, and conclusion validity.

Regarding internal validity, the study was conducted with students from public and private primary and secondary schools. They voluntarily signed up to participate in the online programming workshop. Although we conducted the workshop on four different dates, all occurrences were the same i.e., the treatment with the subjects was the same. The only difference between the students was in relation to the setup they used to be able to follow the workshop and use the application at the same time. While the time of each workshop was approximately 2 hours, it included explanations about SSPOT-VR, evaluation protocols, and participants completing questionnaires. Interaction with the application was typically 30 to 40 minutes. We also ensure that SSPOT-VR usage was performed by all students at the same time and only for the amount of time we stipulate, there were no interruption, and all students present in the workshop were performing only the tasks of the workshop.

As external validity, we were able to evaluate SSPOT-VR with a variety of students. Our goal was to establish SSPOT-VR for usage in a real teaching and learning environment, as if it were an online class for as many different students as possible. As the workshops were conducted online, SSPOT-VR was used without a head-mounted display, so the results can be generalized only to this usage scenario. Another important aspect to consider is that SSPOT-VR was used only by Brazilians students. In order to SSPOT-VR be used by non-Portuguese speakers, it is only necessary to translate the virtual world instructions and provide the options for users to choose.

Aiming to obtain construct validity, we followed instructions proposed by the works we selected as the basis for the acceptance evaluation of SSPOT-VR. We followed the UTAUT model proposed by Venkatesh *et al.* (2003), the EGameFlow scale provided by Fu, Su and Yu (2009), the path model of Bourgonjon *et al.* (2010). For the planning of the evaluation and interpretation of the results, we explained the choices with statistical tests and studies with evaluations similar to SSPOT-VR.

To achieve conclusion validity, the data collected from the user acceptance evaluation were characterized using descriptive statistics and analyzed by factorial analysis. Reliability and data adequacy tests were also performed to confirm the suitability of the data for these tests. All test results are within the considered quality indexes.

To complement the results acquired with the evaluation of user acceptance and use of technology presented herein, in the next section we present an usability evaluation of SSPOT-VR conducted by experts.

4.2 Usability

The International Organization for Standardization (ISO) defines usability as the measure that establishes whether a product can be used by specific users to achieve specific goals with effectiveness, efficiency, and satisfaction in a specific usage context. Effectiveness is the measure of achievement of the initial goals and is evaluated in terms of completion of a task as well as in terms of the quality of the obtained result. Efficiency refers to the effort and resources needed to reach a certain goal, and satisfaction is subjective, it relates to intrinsic factors of each user[6].

Usability evaluations can be conducted using different methods, considering the presence or not of the target audience, through formal or informal user studies, heuristic evaluations with experts, performance predictive models, and so on. Among the usability evaluations methods that do not involve the target audience, evaluations based on heuristics that are carried out by experts have been widely used for assessing the usability of software applications (TORI; HOUNSELL, 2018).

The method initially proposed by Nielsen and Molich (1990) consists of guiding 3 to 5 expert evaluators in the gathering and analysis of violations of a set of 10 heuristics in the software application. Based on the widely accepted approach for user interface evaluation proposed by Nielsen and Molich (1990), Sutcliffe and Gault (2004) presented a set of heuristics and an expert evaluation method specifically for usability evaluation of virtual environments (VE).

The heuristics proposed by Sutcliffe and Gault (2004) are suitable for the evaluation of different nature of VEs. They help to measure the usability levels of the VR application in relation to interaction and the sense of immersion, which are part of the most important attributes for many VR applications. Such heuristics are (SUTCLIFFE; GAULT, 2004):

1. **Natural engagement:** Interaction in the VE should approach the user's expectation of interaction in the real world. Interpreting this heuristic will depend on the naturalness requirement and the user's sense of presence and engagement;

[6] https://www.iso.org/standard/63500.html

2. **Compatibility with the user's task and domain:** The behaviour and look of the VE and of the objects should correspond as closely as possible to the user's expectation of real world;

3. **Natural expression of action:** The representation of the self in the VE should allow the user to act and explore in a natural manner and not restrict normal physical actions. This design quality may be limited by the available devices. If haptic feedback is absent, natural expression inevitably suffers.

4. **Close coordination of action and representation:** The representation of the self and behaviour manifest in the VE should be faithful to the user's actions. Response time between user movement and update of the VE display should be less than 200 ms to avoid motion sickness problems.

5. **Realistic feedback:** The effect of the user's actions on virtual world objects should be immediately visible and conform to the laws of physics and the user's perceptual expectations.

6. **Faithful viewpoints:** The visual representation of the virtual world should map to the user's normal perception, and the viewpoint change by head movement should be rendered without delay.

7. **Navigation and orientation support:** The users should always be able to find where they are in the VE and return to known preset positions. Unnatural actions such as fly-through surfaces may help but these have to be judged in a trade-off with naturalness.

8. **Clear entry and exit points:** The means of entering and exiting from a virtual world should be clearly communicated.

9. **Consistent departures:** When design compromises are used they should be consistent and clearly marked, e.g. cross-modal substitution and power actions for navigation.

10. **Support for learning:** Active objects should be cued and if necessary explain themselves to promote understanding.

11. **Clear turn-taking:** Where system initiative is used it should be clearly signalled and conventions established for turn-taking.

12. **Sense of presence:** The user's perception of engagement and being in a real world should be as natural as possible.

The principles of natural engagement (1), natural expression of action (3) and sense of presence (12) are based on questionnaire-based techniques for assessing the sense of immersion in VR environments. Compatibility with the user's task and domain (2) follows recommendations

for task fit, while heuristics 4 to 7, close coordination, realistic feedback, viewpoints, and navigation support, are based on the taxonomy of VR applications. Heuristics 7–11 maps more directly Nielsen's heuristics for interfaces. Clear turn-taking (11) applies to conversational VEs in which avatars may communicate with the user or when the system takes the initiative (SUTCLIFFE; GAULT, 2004).

Underlying several of the heuristics is the assumption that the VE's role is to represent the real world. Sutcliffe and Gault (2004) explains that in the majority of VEs that is the case. However, there are VEs which represent unnatural worlds. In these cases, heuristics for naturalness need to be interpreted.

4.2.1 Goals

This evaluation was conducted to measure the usability levels of SSPOT-VR, mainly in relation to the intuitiveness of interaction and sense of immersion.

Evaluating the usability of SSPOT-VR makes it possible to gather evidence for fixes and improvements in order to properly promote the expansion of the application, since we planned new levels with new programming content to integrate SSPOT-VR in the future. Besides that, we aim to combine the results of this evaluation with the results of the acceptance and use of technology evaluation, considering that the usability of the application also influences its degree of acceptance.

Therefore, **we aim** to analyze SSPOT-VR **for the purpose** of evaluation, **with respect** to its usability, **from the viewpoint** of experts, **in the context** of the most typical user tasks.

4.2.2 Method

The inspection-based evaluation that we chose enables experts to evaluate SSPOT-VR application based on the twelve specific usability heuristics proposed by Sutcliffe and Gault (2004) to evaluate VR applications. The 12 heuristics are not an assessment tool per se but can be used as a manner of planning the evaluation (TORI; HOUNSELL, 2018).

The method proposed by Sutcliffe and Gault (2004) follows the recommendations provided by Nielsen and Molich (1990) for expert evaluation with an additional step, a technology audit before listing and classifying usability problems. The technology audit establishes the baseline of what the VE can be expected to deliver. The technology audit is carried out in the familiarization period when the evaluator explores the VE and notes the presence or absence of features and any problems associated with the following categories:

- **Operation of the user's presence:** The user may be represented in the virtual world by a simple cursor, hand, or a whole body avatar. The presence may be controlled by a variety of devices, such as 3D mouse, space ball, joystick to pinch gloves and less frequently

whole body immersion suits. For simple navigation, no presence may be necessary; for manipulations, however, a virtual hand is usually necessary.

- **Haptic feedback:** Problems caused by absence of haptic feedback may be observed with complex manipulations and physical tasks.
- **Interactive techniques:** Many VEs implement controls that allow users to fly through VEs to reach and select distant objects by ray-casting. This can be taken further by providing magic 'snap-to' effects so nearby objects automatically jump into the user's hand. These effects can cause usability problems when they are poorly designed.
- **Realistic graphics:** Graphical detail will be important for information displays and for tasks when the system environment is visually complex.

Therefore, the experts start the evaluation by familiarizing themselves with the application. Next, they carry out a set of tasks while listing problems encountered and, finally, use the heuristics to interpret and classify the problems, ranking the severity of the problems. The levels of severity are defined as a four-point scale:

- **Severe:** The problem encountered would make it impossible to complete the task successfully.
- **Annoying:** The problem would disrupt the user's task but most users would learn how to cure the error given an explanation, and some might find a work-around with time.
- **Distracting:** The problem would disrupt the user's tasks but most users would discover the fix relatively quickly given a hint.
- **Inconvenient:** The problem could disrupt the user's task but most users would discover the fix unaided.

The evaluation was voluntarily conducted by five experts accordingly to the method proposed by Sutcliffe and Gault (2004). Experts received documents explaining the use of SSPOT-VR and instructions on conducting the evaluation. They also received the complete list of heuristics, instructions for conducting the technology audit, and the detailed list of tasks to be performed within SSPOT-VR for their evaluation. Experts also had access to a spreadsheet to fill in information about the technology audit and to list the problems found, with description, association to heuristics, and severity ranking.

In the case of this evaluation, even being carried out at a distance, it was possible to provide low cost head-mounted displays and wireless joysticks for the evaluators to conduct the evaluation. Figure 54 reveals the HMD or VR adapter that we used which is commonly known as VR box. Therefore, the impressions and issues encountered by the evaluators have

been identified experiencing SSPOT-VR in an immersive manner with the use of the VR box as an HMD.

Figure 54 – VR box is a low cost head-mounted display with wireless joystick

Source: Author

4.2.3 Results

The experts who conducted the usability heuristic evaluation with SSPOT-VR were four PhD students in Computer Science with more than three years of experience in research on education in Computing, education supported by technology, and mobile learning; and one PhD in Computer Science, with more than 4 years of experience in Computing education. Therefore, this evaluation was voluntarily conducted by 5 experts that are representative subjects of the population. All expert evaluators who performed SSPOT-VR evaluation had previously conducted usability heuristic evaluations; however, they had never used the heuristics proposed by Sutcliffe and Gault (2004) before, until the evaluation of SSPOT-VR. The experts performed the evaluation autonomously and asynchronously according to the instructions that we provided. All of them used a head-mounted display and wireless joystick of the same model, changing only the smartphone model. Data collected from the heuristic usability evaluation were presented and interpreted according to the case studies provided by Sutcliffe and Gault (2004).

The evaluation process includes each evaluator using SSPOT-VR for at least two times. The first for familiarization and the second for collecting violations of the usability heuristics. During the second time, they should follow detailed tasks that simulate the typical usage path that students would follow if they were using the application for the first time. This includes starting the use of SSPOT-VR from the introduction level, where evaluators inspect details on the objects, the environment, and the interaction with the application. Also, moving on to the first challenge level, where evaluators are inserted into the application's narrative and tasked to program the robot named Mark using blocks. Once again, inspecting inspect details of each object, the environment, and the way in which the user interaction occurs with these objects and with the environment.

The first time evaluators use SSPOT-VR, also known as the familiarization period or technology audit, they registered their impressions of SSPOT-VR according to the user's presence; haptic categories feedback; interactive techniques, and realistic graphics. The union

of the impressions of the five evaluators led us to produce the following technology audit of SSPOT-VR:

- **Operation of the user's presence:** The user's presence within the virtual world is a circular reticle pointer that automatically enlarges its size when it is positioned over an object that allows interaction. The reticle pointer is supported by the gaze-based interaction. Users also use the wireless joystick to perform manual confirmation of actions. The presence can be enhanced with the use of avatars or virtual hands, as well as whether the virtual world is able to track user's movement beyond the 3 degrees of freedom (3-DoF).
- **Haptic feedback:** Haptic feedback was substituted by the combination of the reticle pointer, gaze-based interaction, and manual confirmation performed with a wireless joystick. The application provides an efficient feedback for interaction, e.g., when users take a block or hit a button. All actions performed in the virtual world are received by the application and generate results.
- **Interactive techniques:** The interactive objects are manipulated through the combination of the gaze-based interaction and manual confirmation performed with a wireless joystick. Interaction techniques are suitable for the hardware used. On a certain level, the virtual world allows for exploration and discovery. The controls are well designed and there are visual cues that guide the user through the virtual world. Although, realism would be positively affected if information were presented in the form of subtitles, narration, or voice interaction.
- **Realistic graphics:** The application provides high fidelity graphics. The virtual world and instructions can be completely visualized with quality from the user's point of view. Since the high fidelity graphics synergies with the visual and narrative, the virtual world provides the feeling of immersion. However, the set of colors and the font style used to present information may not be suitable for all types of users.

In summary, the compilation of the technology audit conducted by the five evaluators indicates that SSPOT-VR is a virtual world that provides high fidelity graphics, immersion, and interaction with its environment and objects through the combination of the reticle pointer, gaze-based interaction, and an external wireless joystick that confirms users' actions. Considering the low cost setup we used to develop SSPOT-VR, haptic feedback is not available, as well as advanced user movement tracking.

When the expert evaluators were using SSPOT-VR for the second time, they registered the violations of the usability heuristics proposed by Sutcliffe and Gault (2004). As SSPOT-VR was thoroughly explored, ten different problems were encountered leading to the analysis of these usability problems with reference to the heuristics. Chart 6 summarizes the problems encountered with SSPOT-VR between five of the twelve heuristics along with the rankings that

provide a summative evaluation of the problems with the virtual world (NIELSEN; MOLICH, 1990).

Chart 6 – Heuristics rating and interpretation of problems encountered

Heuristic	Rating	Problems
1. Natural engagement	6	A lot of textual information is presented at once and the colors of certain objects can cause confusion, as well as the font style and the position of some of the instructional panels.
2. Compatibility with the user's task	2	The compiler used to process the blocks programmed by the users might fail given a specific case and the compilation error messages are sometimes not clear enough.
3. Natural expression of action	2	The usage of the joystick is not intuitive and need to be explained to users prior to using the application; moving the head and neck constantly can cause fatigue.
4. Close coordination	0	N/A
5. Realistic feedback	0	N/A
6. Faithful viewpoints	0	N/A
7. Navigation and orientation support	0	N/A
8. Clear entry and exit points	4	The application does not make it explicit enough that the user has completed the level.
9. Consistent departures	0	N/A
10. Support for learning	1	Objects that allow interaction are not highlighted, the widening of the gaze reticle when it is positioned over the object could be not enough to alert users.
11. Clear turn-taking	0	N/A
12. Sense of presence	0	N/A

Analyzing Chart 6, we can notice that evaluators encountered ten different issues related to five different heuristics. Since problems can be associated with more than one heuristic, the attribution is assigned to the heuristic which explains the error most directly, followed by supplementary explanations (SUTCLIFFE; GAULT, 2004).

It is also possible to verify that most of the encountered problems are related to the heuristic of natural engagement (1). Accordingly to the evaluators, these problems were caused by the manner we adopted to provide instructions to users. Four different problems were found in total that are associated with this heuristic. Two of these problems were identified by two different evaluators, explaining the rating equal to six. A total of two out of five evaluators mentioned problems with the amount of textual information and position of the instruction panels, while one evaluator mentioned that color choices can confuse users; one evaluator also mentioned that the font style we used could be difficult to read for some users.

The next most common problem with SSPOT-VR is related to the heuristic of clear entry and exit points to the virtual world (8). Evaluators state that the application does not make it explicit enough that the tasks of that level are complete. This same issue was described by four different evaluators, explaining the rating equal to four for this heuristic.

Other issues were also found less frequently. One evaluator identified a problem with the operation of the algorithm's compiler, which may fail given a specific case of usage of blocks. In other words, there is no compilation problem with the block's algorithm, but the compiler indicates a compilation problem, being necessary to reset the algorithm and create a new one from scratch.

Also associated with the algorithm's compiler, one evaluator indicated that the messages

the compiler displays when the algorithm created has an error could be more meaningful. In addition to the reticle pointer that automatically enlarges itself, one evaluator thought it necessary to highlight the objects that allow interaction, as well as one evaluator identified that the joystick needs a prior explanation to be used efficiently. Finally, one evaluator believes that the necessary movement with the head and neck to use SSPOT-VR can be excessive. All ten problems were identified with inconvenient, distracting, or annoying severity.

The ten different identified issues are discussed next, along with the association with a feature category, severity rating, and design change.

4.2.4 Discussion

Analyzing the usability problems found by the expert evaluators, it is possible to verify that although we have based the creation of SSPOT-VR on guidelines for the development of VR applications, there are improvements to be made and problems to be fixed. This is necessary to ensure that SSPOT-VR can be used by specific users to achieve specific goals with effectiveness, efficiency, and satisfaction, in a specific usage context[7].

The ten problems found by evaluators were then interpreted with the heuristics to evaluate the quality of presence, which is given by problems related to the heuristics 1 and 12, with contributions from heuristics 2 to 6, while design problems are accountable from heuristics 2 to 11. Considering that the problems found in SSPOT-VR are associated with the heuristics natural engagement (1), compatibility with the user's task (2), natural expression of action (3), clear entry and exit points (8), and support for learning (10), it is possible to verify that these heuristics are more related to design problems than to the quality of presence provided by the application.

The problems can also be attributed to the categories of features for virtual environments defined by Sutcliffe and Gault (2004) for an objective discussion, together with the severity rating, and suggestion for design change, as illustrated in Chart 7.

Along with the categorization of issues found by feature, the severity of the identified issues is provided, ranging from a poor design with a severe impact that can result in a task failure to a minor issue likely to be curable by training. This ranking reflects the evaluator's judgment about the severity of those errors, on a four-point scale (SUTCLIFFE; GAULT, 2004): (1) inconvenient; (2) distracting; (3) annoying; (4) severe.

Although subjective, SSPOT-VR problems associated with its graphics are characterized by the use of colors that can cause confusion for some users. Colors can be associated with actions and feelings, making the usage of colors necessary accordingly to theories that establish standards for intuitiveness in interfaces for children and teenagers. Regarding the font style used, evaluators recommend using a simple *sans serif* font style to support reading. These graphics aspects, in addition to being subjective, depend on the hardware being used, considering the

[7] https://www.iso.org/standard/63500.htm

Chart 7 – Classification of problems encountered with severity ratings and suggested design improvements

Feature	Problem description	Severity rating	Design change
Graphics	The colors of certain objects can cause confusion as well as the font style.	Inconvenient	Usage of colors accordingly to theories that establish standards for intuitiveness; adopt a sans serif font style that supports reading.
Presence	Joystick usage could be not intuitive; head movement can cause fatigue; the presentation of textual instructions can be seen as unnatural.	Distracting	Present how to use the joystick within the application; decrease neck movement angles; add other methods to communicate with users, such as through narration.
Interaction	Objects that allow interaction are not highlighted; it is not clear when a level is complete; compiler might fail given a specific case and compiler error messages are not sufficiently clear.	Annoying	Fix any bug with the compiler; give more meaningful feedback about compilation errors; clearly indicate that the level has ended or that the user is ready to continue; highlight objects that allow interaction.
Environmental features	N/A	N/A	N/A
Controls	N/A	N/A	N/A
Hardware	N/A	N/A	N/A

quality and resolution of the mobile device's display and its processing power. The graphics issues were ranked as inconvenient, meaning that the problem could disrupt the user's task but most users would use the application without assistance.

Regarding presence, evaluators found the amount of text on instructional panels to be excessive, as well as the position of certain instructional panels could be inadequate. These factors negatively influence the users' sense of presence as it differs from how information is acquired in the real world. That is the reason it is suggested by evaluators to insert voice commands and narration.

Problems with instructional panels can also be associated with the movement of user's neck and head. Objects in the virtual world should require minimal effort to be seen and acted upon. Furthermore, the usage of wireless joystick can also affect the users' sense of presence, considering that it is an external controller that needs explanation to be used. Therefore, these problems were ranked as distracting. They can disrupt the user's tasks, however most users would discover the fix relatively quickly given a hint. This indicates that SSPOT-VR method of provide instructions works, although it can be improved to offer a better experience.

The interaction problems with SSPOT-VR received the highest level of severity among the encountered problems, the annoying level. These problems would disrupt the user's task but most users would learn how to cure the error given an explanation, and some might find a work-around with time. The error that can happen with the compiler, although it was found by only one of the evaluators, is one of the issues that most affects the use of the application, as it is a failure, not associated with any user perception. Also related to the compiler, compilation error messages can provide more meaningful feedback to the users. We have attempted to leave users with minimal feedback to promote critical thinking and problem solving. Evaluators have indicated it is a point that needs improvement.

Another significant problem indicated by the evaluators is that the application does not make it explicit enough that users have completed the level or are ready to continue. Although a message is given when completing the programming task, this message is not presented in the way that evaluators deem most efficient. Furthermore, just using the reticle pointer to indicate that an object allows interaction may be insufficient for certain users, according to the evaluators it is also important to highlight objects that allow interaction. As there was no problem encountered that would make it impossible to complete the tasks successfully, no feature class was ranked as severe. In general, the usability problems with SSPOT-VR do not prevent its use, although they indicate how to prioritize certain areas for redesign in the next version, accordingly to the normal practice of an heuristic evaluation (NIELSEN; MOLICH, 1990).

4.3 Final Remarks

Establishing evaluation methods that are easy to apply, broad within a certain context, and that allow, at least partially, the automation of the process, is still a challenge in the VR literature (TORI; HOUNSELL, 2018). Nunes, Machado and Moraes (2014) verified that 46% of the works published in the Symposium on Virtual and Augmented Reality (SVR)[8] that presented VR applications for healthcare purposes did not present any type of evaluation. Therefore, it is possible to verify the low frequency for conduction of evaluations with VR applications, even when it is in the healthcare area.

The results of the user acceptance and use of technology evaluation reveal that students accept to use SSPOT-VR. The subjects of the evaluation believe that they are able to use SSPOT-VR in the classroom, and that its usage supports the learning of programming, therefore, students would adopt it for usage in their daily school life.

After analyzing the user acceptance and use of SSPOT-VR established on the three studies we based on, we sought to highlight factors that influence students and are determinant in the acceptance of applications such as SSPOT-VR, as well as for its use. The analysis of the main components shows a relationship between characteristics extracted from the data, aiming at reducing factors and identifying the number of those that are able to capture most of the variation in the data. This reduction enables us to realize that the user acceptance and use of SSPOT-VR depends on the learning opportunities that the application provides, as well as the affection students feel when using the application. The better the learning opportunity, or how it is perceived by students, and how the affection of students will subjectively act, the greater the chance that SSPOT-VR will be accepted by students for use.

Regarding the heuristic usability evaluation that five experts evaluators conducted with SSPOT-VR, there are no usability issues with SSPOT-VR that have high severity; however, there are improvements regarding its graphics, sense of presence, and interaction methods. Even so,

[8] https://www.computer.org/csdl/proceedings/svr/2014/12OmNynsbxj

SSPOT-VR proves to be a viable application to support programming learning, as it acquires the attention of its users. Our results also reveal that the combination of mobile applications, virtual reality, block-based programming, and storytelling, could characterize an efficient method for the teaching and learning of programming for children and teenagers.

CHAPTER

5

CONCLUSIONS

Programming skills has been described as the new Latin of the modern school curriculum, as a type of mental sharpener for developing minds (SLEEMAN, 1986). It benefits students of different ages and with different interests. It is required in several undergraduate and graduate programs and in different certificate programs. Programming has even become common in the primary and secondary school curriculum of several countries worldwide (KALELIOGLU, 2015). It has also become an important skill for everyone living in today's increasingly digital society (JOHNSON *et al.*, 2016; ADAMS *et al.*, 2017; ADAMS *et al.*, 2018; UNESCO, 2018).

While the benefits associated with programming skills are broad, the teaching and learning of programming has been facing many problems, difficulties, and challenges. These issues have been widely discussed (CHAKRAVERTY; CHAKRABORTY, 2020; AISSA *et al.*, 2020; CHEAH, 2020; NOURI *et al.*, 2020; FRANCISCO, 2021). It is possible to categorize the issues as: (i) lack of a universal definition of what to learn; (ii) students' motivation or ability to problem-solving; (iii) students' competence in learning and applying programming concepts; (iv) instructors adoption of static and traditional teaching materials, and (v) school or university curriculum.

At the same time, research on solutions for the teaching and learning of programming have been adopting different approaches and technologies, such as: (i) block-based programming languages or environments, that present a lower learning barrier compared to textual programming environments (BAU *et al.*, 2017; GROVER; BASU, 2017; WEINTROP, 2019; BALL *et al.*, 2019); (ii) mobile learning applications, that add ubiquity to learning, allowing the learning activity to take place anywhere at any time [1,2](TILLMANN *et al.*, 2012; JORDINE; LIANG; IHLER, 2014), and (iii) different setups of VR (VINCUR *et al.*, 2017; TANIELU *et al.*, 2019; SEGURA *et al.*, 2020), that brings immersion to the teaching and learning process, involving students in stories, experiences, and virtual worlds that stimulate their senses in the same way as

[1] https://www.scratchjr.org
[2] https://grasshopper.app

the real world.

According to Cheah (2020), the issues with the teaching and learning of programming persist until present day. Students continue to struggle to learn to program for reasons that the hypothesis should not just be restricted to cognitive aspects. It is necessary to recognize that part of the problems related to programming learning may have its origin in the teaching and learning practices (FRANCISCO, 2021).

Despite several research related to block-based programming languages, mobile learning applications, and VR applications, except for SSPOT-VR, there is a lack of studies in the literature that brings these three elements together in the same application. Besides that, immersive VR has been seldom used to support the teaching and learning of programming, even less frequently when it comes to mobile immersive VR (AVELLAR; BARBOSA, 2019).

A small number of research outside of Brazil adopted these technologies to support the teaching and learning of programming, as in the work of Vincur *et al.* (2017), that uses immersive VR and BBP with a sophisticated VR setup, allowing teenagers to solve Code.org challenges in a game-like context; the application proposed by Tanielu *et al.* (2019) uses immersive VR to teach OOP to undegraduate students and does not provide details on its VR setup, and in the work of Segura *et al.* (2020), an immersive VR application with a sophisticated setup is presented, using an adaptation of BBP to teach fundamental programming concepts to children, teenagers and adults, in a game-like context.

Compared to related work, SSPOT-VR: (i) had its educational content based on complete and specific educational theory for teaching programming and allows expansion according to the evolution of the content presented in the documents of SBC (2019) and Raabe, Brackmann and Campos (2018); (ii) it requires a simple and affordable VR setup, based on the software and hardware platform of mobile devices, with the option to have an immersive or non-immersive experience, and (iii) it uses block-based programming principles and storytelling elements to work together with immersive VR to engage and motivate students, as well as to present programming content in a joyful manner.

In contrast with SSPOT-VR, similar applications do not detail the process of planning and development of the virtual world they present, such as details on how they selected the approached content of programming and the teaching and learning methods, nor how they developed the software. On the other hand, we present all the educational and technical aspects associated with SSPOT-VR, together with detailed explanations of each level built.

The main objective of this Master's research is to answer whether or not students from primary and secondary public and private schools would adopt a mobile immersive VR application for supporting the teaching and learning of programming. In addition to that, we aimed to enable students to have a positive experience with the theoretical concept and the construction of algorithms, as recommended by SBC (2019).

As presented in Chapter 4, we evaluated the application with the target audience. According to the results, SSPOT-VR is accepted for use and would be adopted for use by children, teenagers, and young adults. Our subjects' ages ranged from 6 to 19 years old. Students judged that SSPOT-VR presented enjoyable learning activities. Students believe that applications similar to SSPOT-VR are able to teach them, and after using SSPOT-VR they want to learn more about programming. We also discovered that the user acceptance and use of SSPOT-VR strongly depends on the learning opportunities that the application provides, as well as the affection students feel about the application.

We also conducted a usability evaluation of SSPOT-VR with experts evaluators. The experts indicated improvements accordingly to usability heuristics for VR applications. The usability issues encountered are not of major impact, however they can provide improvement. The results of the evaluations support the notion that the chosen guidelines for programming teaching and documents for VR applications development, together with our additions, were adequate for creating SSPOT-VR.

SSPOT-VR can be adopted as a free and useful technological resource to assist instructors and students in the process of teaching and learning of programming. When students are exposed to basic arithmetic operations in school, instructors can also present the basic definition of algorithms. This enables students to use and follow instructions from different sources to develop algorithms to solve different types of problems, using both natural languages and visual languages (SBC, 2019).

By using SSPOT-VR students empower themselves with different computing and technology elements. The use of mobile devices, the immersive experience with VR, a visual programming language and the contact with the concept of algorithm support the presence and continuity of the fundamentals of programming skills in students' lives. This introduction to programming also enables them to better understand the world, to have autonomy, flexibility, resilience, and creativity (SBC, 2019).

Therefore, through the usage of ICTs, virtual worlds such as SSPOT-VR are being capable to increase the sense of closeness perceived by students. Even though it is not yet possible to fully replace formal education, such as the experience of physically manipulate an object, digital learning technologies can minimize the distance effects that take place during the learning process. Today, ICTs are being applied in all types of education, instructors and students are discovering that virtual resources can be a major support for face-to-face activities (TORI, 2010a).

5.1 Research Contributions

The main contributions of this Master's research are the following:

- **Characterization of the state-of-the-art on VR and AR applications for supporting the teaching and learning of programming:** we conducted a systematic mapping study for the characterization of the current scenario of VR, AR, and/or MR applications in the teaching and learning of programming. This MS helped us to understand the research area as well as to identify trendings and lacks of research. The results revealed that both VR and AR have been supporting the teaching and learning of programming. Results also showed a lack of research on the usage of mobile VR applications and mobile immersive VR applications.
- **Creation of a mobile VR application for the teaching and learning of programming:** we created a mobile immersive VR application for supporting the teaching and learning of programming to children and teenagers using BBP principles and storytelling elements, named SSPOT-VR, a Space Station for Programming Training in Virtual Reality. SSPOT-VR integrates methods for the teaching and learning of programming and the simulated experience of a digitally created world.
- **Evaluation of the proposed application:** we evaluated SSPOT-VR with the target audience with the goal of verify the acceptance and usage of SSPOT-VR among children and teenagers. We also evaluated the usability of SSPOT-VR with experts. Preliminary results provided evidence that our virtual world is accepted to support the teaching and learning programming, with educational potential, and adequate usability.

5.2 Research Limitations

The main limitations related to the work conducted in this Master's research are described as follows:

- **Number of different levels of the virtual world:** in order to answer our research question, we developed a fully functional introduction level and first challenge level for the SSPOT-VR application. We intend to develop more levels with new programming concepts to integrate the application in the future.
- **Evaluation in person and using head-mounted display:** due to COVID-19, we were unable to conduct the evaluation within the classroom, which means that no head-mounted displays were used during the evaluation that we conducted with children and teenagers. On the other hand, the usability evaluation addressed the usage of head-mounted displays.
- **Evaluation focused on the learning aspect:** in this work, the evaluation goals of SSPOT-VR included the verification of the interest of the target audience in its use and whether or not they would accept it for use while thinking that they can learn something. We also focused on identifying whether the application was created following usability principles. We did not directly evaluated the effectiveness of programming learning using SSPOT-VR.

5.3 Future Work

We can provide different possibilities for future directions of the work conducted in this Master's research, that are summarized as follows:

- **Develop more levels for the teaching of new programming concepts:** we created an application which usage can be expanded through primary and secondary education. We intend to develop more levels for SSPOT-VR, approaching new concepts of programming, such as the second, third, fourth, and fifth challenge presented in Chapter 3.
- **Conduction of more evaluations:** in addition to the evaluation for usage and acceptance conducted with the target audience, and the usability evaluation conducted with mobile learning applications experts, we plan to conduct a learning effectiveness evaluation with SSPOT-VR. In this context, we can conduct an evaluation with pre-tests and post-tests that compares traditional teaching methods and other applications that support the teaching and learning programming with the SSPOT-VR application.
- **Add interaction through voice commands:** the virtual world instructions and its narrative could be delivered through multiple methods including narration. The more delivery methods are available, the better the chances students will follow along the story and the programming activities.
- **Add collaborative programming:** collaborative programming is another resource to support the teaching and learning of programming that has been adopted by instructors to teach programming to students of different ages. Since Unity3D enable us to create cross-platform network applications, adding this feature to SSPOT-VR increases its educational and motivational potential.
- **Add augmented reality features:** we created SSPOT-VR using the Unity 3D game engine and therefore it is possible to add AR features in VR applications and vice-versa. SSPOT-VR would be, in this case, the only mobile application that supports teaching and learning of programming with resources from both VR and AR.

5.4 Resulting Publications

The publications resulting from the activities conducted during this Master's research are as follows:

- AVELLAR, G. M. N.; BARBOSA, E. F. Virtual and augmented reality in the teaching and learning of programming: A systematic mapping study. In: **Brazilian Symposium on Computers in Education (*Simpósio Brasileiro de Informática na Educação* - SBIE)**, 2019. ISSN 2316-6533. Available: <https://www.br-ie.org/pub/index.php/sbie/article/view/8774>.

Other works indirectly related to this Master's work have been published:

- AVELLAR, G. M. N.; SILVA, R. F. d.; SCATALON, L. P.; ANDRADE, S. A.; DELAMARO,M. E.; BARBOSA, E. F. Integration of software testing to programming assignments: An experimental study. In: **2019 IEEE Frontiers in Education Conference (FIE)**, 2019. ISSN 2377-634X. Available: <https://ieeexplore.ieee.org/abstract/document/9028519/>
- FIORAVANTI, M. L.; OLIVEIRA, R. D. G.; AVELLAR, G. M. N.; OLIVEIRA, C. D. de; BARBOSA, E. F. An analysis of projectedu: A mobile learning application for software project management education. In: **Learning and Collaboration Technologies. Ubiquitous and Virtual Environments for Learning and Collaboration**, 2019. ISBN 978-3-030-21817-1. Available: <https://link.springer.com/chapter/10.1007/978-3-030-21817-1_4>

BIBLIOGRAPHY

ACM, A.; SOCIETY, I. C. **Computing Curricula 2005: The Overview Report**. 2005. Access in: 2021-04-20. Available: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf>. Citation on page 29.

ACM, I.; SOCIETY, I. C. **Computing Curricula 2020: Paradigms for Global Computing Education**. 2020. Access in: 2021-04-20. Available: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf>. Citation on page 29.

ADAMS, S.; BROWN, M.; DAHLSTROM, E.; DAVIS, A.; DEPAUL, K.; DIAZ, V.; POMERANTZM, J. **NMC Horizon Report: 2018 Higher Education**. Louisville, CO: EDUCAUSE, 2018. Available: <https://library.educause.edu/~/media/files/library/2018/8/2018horizonreport.pdf>. Citations on pages 23 and 135.

ADAMS, S.; CUMMINS M., D. A.; FREEMAN, A.; GIESINGER, C. H.; ANANTHANARAYANAN, V. **NMC Horizon Report: 2017 Higher Education**. Austin, Texas: The New Media Consortium, 2017. Available: <http://cdn.nmc.org/media/2017-nmc-horizon-report-he-EN.pdf>. Citations on pages 23 and 135.

AISSA, M.; AL-KALBANI, M.; AL-HATALI, S.; BINTOUQ, A. Novice learning programming languages in omani higher education institution (nizwa university) issues, challenges and solutions. In: AL-MASRI, A. N.; AL-ASSAF, Y. (Ed.). **Sustainable Development and Social Responsibility—Volume 2**. Cham: Springer International Publishing, 2020. ISBN 978-3-030-32902-0. Citations on pages 35, 36, and 135.

AIVALOGLOU, E.; HERMANS, F. How kids code and how we know: An exploratory study on the scratch repository. In: **Proceedings of the 2016 ACM Conference on International Computing Education Research**. New York, NY, USA: ACM, 2016. (ICER '16), p. 53–61. ISBN 978-1-4503-4449-4. Available: <http://doi.acm.org/10.1145/2960310.2960325>. Citation on page 25.

ALLEN, J. M.; VAHID, F.; DOWNEY, K.; EDGCOMB, A. D. Weekly programs in a cs1 class: Experiences with auto-graded many-small programs (msp). In: **2018 ASEE Annual Conference & Exposition**. Salt Lake City, Utah: ASEE Conferences, 2018. Https://peer.asee.org/31231. Citation on page 34.

ALMAIAH, M. A.; ALAMRI, M. M.; AL-RAHMI, W. Applying the utaut model to explain the students' acceptance of mobile learning system in higher education. **IEEE Access**, IEEE, 2019. Citation on page 109.

ARANHA, T. S. e Tainá Medeiros e Handerson Medeiros e Ranyer Lopes e E. Ensino-aprendizagem de programação: uma revisão sistemática da literatura. **Revista Brasileira de Informática na Educação**, v. 23, 2015. ISSN 2317-6121. Available: <https://www.br-ie.org/pub/index.php/rbie/article/view/2838>. Citation on page 34.

ASHTARI, N.; BUNT, A.; MCGRENERE, J.; NEBELING, M.; CHILANA, P. K. Creating augmented and virtual reality applications: Current practices, challenges, and opportunities. In: **Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems**. Association for Computing Machinery, 2020. (CHI '20). ISBN 9781450367080. Available: <https://doi.org/10.1145/3313831.3376722>. Citation on page 55.

AVELLAR, G. M. N.; BARBOSA, E. F. Virtual and augmented reality in the teaching and learning of programming: A systematic mapping study. **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)**, 2019. ISSN 2316-6533. Available: <https://www.br-ie.org/pub/index.php/sbie/article/view/8774>. Citations on pages 26, 61, 73, 95, 96, 97, and 136.

AZUMA, R.; BAILLOT, Y.; BEHRINGER, R.; FEINER, S.; JULIER, S.; MACINTYRE, B. Recent advances in augmented reality. **IEEE Computer Graphics and Applications**, v. 21, n. 6, p. 34–47, Nov 2001. ISSN 0272-1716. Citation on page 56.

BACHMANN, F.; BASS, L.; BUHMAN, C.; COMELLA-BORDA, S.; LONG, F.; ROBERT, J.; SEACORD, R.; WALLNAU, K. Technical concepts of component-based software engineering. **CMU/SEI-2000-TR-008**, Pittsburgh, PA, 2000. Available: <http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=5203>. Citation on page 39.

BALANSKAT, A.; ENGELHARDT, K. **Computing our future: Computer programming and coding Priorities, school curricula and initiatives across Europe**. European Schoolnet. EUN Partnership AIBSL, 2015. Access in: 2021-03-07. Available: <http://www.eun.org/documents/411753/817341/Computing+our+future_final_2015.pdf/d3780a64-1081-4488-8549-6033200e3c03>. Citations on pages 31 and 33.

BALL, T.; CHATRA, A.; HALLEUX, P. de; HODGES, S.; MOSKAL, M.; RUSSELL, J. Microsoft makecode: Embedded programming for education, in blocks and typescript. In: **Proceedings of the 2019 ACM SIGPLAN Symposium on SPLASH-E**. Association for Computing Machinery, 2019. (SPLASH-E 2019). ISBN 9781450369893. Available: <https://doi.org/10.1145/3358711.3361630>. Citations on pages 46 and 135.

BARAN, E. A review of research on mobile learning in teacher education. **Educational Technology Society**, v. 17, p. 17–32, 10 2014. Citation on page 44.

BAU, D. Droplet, a blocks-based editor for text code. **J. Comput. Sci. Coll.**, Consortium for Computing Sciences in Colleges, USA, v. 30, n. 6, p. 138–144, Jun. 2015. ISSN 1937-4771. Available: <http://dl.acm.org/citation.cfm?id=2753024.2753052>. Citations on pages 24, 26, 41, and 43.

BAU, D.; BAU, D. A.; DAWSON, M.; PICKENS, C. S. Pencil code: Block code for a text world. In: **Proceedings of the 14th International Conference on Interaction Design and Children**. New York, NY, USA: ACM, 2015. (IDC '15), p. 445–448. ISBN 978-1-4503-3590-4. Available: <http://doi.acm.org/10.1145/2771839.2771875>. Citations on pages 40 and 41.

BAU, D.; GRAY, J.; KELLEHER, C.; SHELDON, J.; TURBAK, F. Learnable programming: Blocks and beyond. **Commun. ACM**, ACM, New York, NY, USA, v. 60, n. 6, p. 72–80, May 2017. ISSN 0001-0782. Available: <http://doi.acm.org/10.1145/3015455>. Citations on pages 39, 42, 43, 98, and 135.

BIMBER, O.; RASKAR, R. Modern approaches to augmented reality. In: **ACM SIGGRAPH 2006 Courses**. New York, NY, USA: ACM, 2006. (SIGGRAPH '06). ISBN 1-59593-364-6. Available: <http://doi.acm.org/10.1145/1185657.1185796>. Citation on page 57.

BOURGONJON, J.; VALCKE, M.; SOETAERT, R.; SCHELLENS, T. Students' perceptions about the use of video games in the classroom. **Computers Education**, 2010. ISSN 0360-1315. Available: <https://www.sciencedirect.com/science/article/pii/S0360131509003121>. Citations on pages 108, 109, 110, 113, 116, 121, and 123.

BRASIL. **Base Nacional Comum Curricular**. Brasília: Ministério da Educação, 2018. Citations on pages 24 and 32.

CAMPOS, F. R.; DIAS, R. A. **Currículo de Referência e Itinerário Formativo em Tecnologia e Computação para o Ensino Médio**. São Paulo: Centro de Inovação para a Educação Brasileira (CIEB), 2020. Ebook in pdf. Citations on pages 29, 36, and 87.

CARDOSO, A.; MACKENZIE, I. F.; KIRNER, C.; TORI, R. Development of educational resources with virtual and augmented reality: Challenges and perspectives. In: **2017 XLIII Latin American Computer Conference (CLEI)**. [S.l.: s.n.], 2017. p. 1–6. Citation on page 26.

CARVALHO, M. F.; AGUIAR, Y. P. C.; DANTAS, V. F. Ensino da estrutura de repetição for em python com realidade aumentada através do aurasma. Universidade Federal da Paraíba, 2017. Citation on page 65.

CAUDELL, T. P.; MIZELL, D. W. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In: **Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences**. [S.l.: s.n.], 1992. ii, p. 659–669 vol.2. Citation on page 56.

CHAKRAVERTY, K. S.; CHAKRABORTY, P. Tools and techniques for teaching computer programming: A review. **Journal of Educational Technology Systems**, v. 49, n. 2, p. 170–198, 2020. Citations on pages 34, 37, and 135.

CHANDRAMOULI, M.; HEFFRON, J. A Desktop VR-based HCI framework for programming instruction. **ISEC 2015 - 5th IEEE Integrated STEM Education Conference**, p. 129–134, 2015. ISSN 2330-331X. Citations on pages 29, 65, and 68.

CHANDRAMOULI, M.; ZAHRAEE, M.; WINER, C. A fun-learning approach to programming: An adaptive Virtual Reality (VR) platform to teach programming to engineering students. **IEEE International Conference on Electro Information Technology**, p. 581–586, 2014. ISSN 21540373. Citations on pages 24 and 65.

CHEAH, C. S. Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. **Contemporary Educational Technology**, v. 12, 2020. Available: <https://www.cedtech.net/article/factors-contributing-to-the-difficulties-in-teaching-and-learning-of-computer-programming-a-8247>. Citations on pages 24, 34, 36, 135, and 136.

COCHRANE, T. Mobile vr in education: From the fringe to the mainstream. In: ____. [S.l.: s.n.], 2018. ISBN 9781522554707. Citation on page 47.

COOPER, S.; DANN, W.; PAUSCH, R. Alice: a 3-d tool for introductory programming concepts. **Journal of Computing Sciences in Colleges**, v. 15, n. 5, p. 107–116, 2000. Citation on page 39.

COOPER, S.; DANN, W.; PAUSCH, R.; PAUSCH, R. Teaching objects-first in introductory computer science. In: **Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education**. New York, NY, USA: ACM, 2003. (SIGCSE '03), p. 191–195. ISBN 1-58113-648-X. Available: <http://doi.acm.org/10.1145/611892.611966>. Citation on page 42.

CORNEY, M.; TEAGUE, D.; THOMAS, R. N. Engaging students in programming. In: **Proceedings of the Twelfth Australasian Conference on Computing Education - Volume 103**. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2010. (ACE '10), p. 63–72. ISBN 978-1-920682-84-2. Available: <http://dl.acm.org/citation.cfm?id=1862219.1862230>. Citation on page 25.

CRAIG, A.; COLDWELL-NEILSON, J.; GOOLD, A.; BEEKHUYZEN, J. A review of e-learning technologies: Opportunities for teaching and learning. In: . [S.l.: s.n.], 2012. v. 1. Citation on page 32.

CSTA; ISTE. **Computational Thinking Teacher Resources**. 2011. Access in: 2021-03-07. Available: <https://cdn.iste.org/www-root/ct-documents/ct-teacher-resources_2ed-pdf.pdf?sfvrsn=2>. Citation on page 30.

CUSHING, A.; GANTZ, J. F. Skills requirements for tomorrow's best jobs: Helping educators provide students with skills and tools they need. p. 4–15, 2013. ISSN 0277786X. Available: <https://news.microsoft.com/download/presskits/education/docs/IDC_101513.pdf>. Citation on page 34.

DALE, E. Audio-visual methods in teaching. The Dryden Press, 1969. Citation on page 25.

DALGARNO, B.; LEE, M. J. W. What are the learning affordances of 3-d virtual environments? **British Journal of Educational Technology**, 2010. Available: <https://bera-journals.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8535.2009.01038.x>. Citation on page 95.

DÍAZ, G.; ALBARRÁN, M.; CALERO, P. A. Role-play virtual environments: Recreational learning of software design. In: DILLENBOURG, P.; SPECHT, M. (Ed.). **Times of Convergence. Technologies Across Learning Contexts**. [S.l.: s.n.], 2008. p. 27–32. ISBN 978-3-540-87605-2. Citation on page 65.

DUNCAN, C.; BELL TIM ANDTANIMOTO, S. Should your 8-year-old learn coding? In: **Proceedings of the 9th Workshop in Primary and Secondary Computing Education**. New York, NY, USA: ACM, 2014. (WiPSCE '14), p. 60–69. ISBN 978-1-4503-3250-7. Available: <http://doi.acm.org/10.1145/2670757.2670774>. Citations on pages 31 and 32.

ECONOMIDES, A. A. Requirements of mobile learning applications. **International Journal of Innovation and Learning**, v. 5, p. 457–479, 04 2008. Citations on pages 44 and 47.

FIGUEIREDO, M.; CHACÓN, M.-Á.; GONÇALVES, V. Learning programming and electronics with augmented reality. In: ANTONA, M.; STEPHANIDIS, C. (Ed.). **Universal Access in Human-Computer Interaction. Users and Context Diversity**. Cham: Springer International Publishing, 2016. p. 57–64. ISBN 978-3-319-40238-3. Citation on page 65.

FOG, K.; BUDTZ, C.; MUNCH, P.; BLANCHETTE, S. The four elements of storytelling. In: ____. **Storytelling: Branding in Practice**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. ISBN 978-3-540-88349-4. Available: <https://doi.org/10.1007/978-3-540-88349-4_2>. Citations on pages 47, 48, 49, and 76.

FRANCISCO, J. H. B. e Antonio Carlos de. Percepção de docentes que lecionam programação de computadores quanto à formação pedagógica. **Revista Brasileira de Informática na Educação**, v. 29, p. 133–159, 2021. ISSN 2317-6121. Available: <https://www.br-ie.org/pub/index.php/rbie/article/view/v29p133>. Citations on pages 24, 30, 35, 37, 135, and 136.

FU, F.-L.; SU, R.-C.; YU, S.-C. Egameflow: A scale to measure learners' enjoyment of e-learning games. **Computers Education**, v. 52, n. 1, p. 101–112, 2009. ISSN 0360-1315. Available: <https://www.sciencedirect.com/science/article/pii/S0360131508001024>. Citations on pages 53, 108, 109, 110, 113, 116, 121, 122, and 123.

GARTNER, G. **Top Trends in the Gartner Hype Cycle for Emerging Technologies, 2017**. 2017. Access in: 2019-01-31. Available: <https://www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/>. Citation on page 26.

GLINERT, E. Towards second generation interactive graphical programming environments. In: **Proceedings of IEEE Workshop onVisual Language. IEEE CS Press, Silver Spring, MD**. [S.l.: s.n.], 1986. p. 61–70. Citation on page 39.

GOOGLE. **Google Cardboard**. 2020. Access in: 2020-08-08. Available: <https://vr.google.com/cardboard/>. Citations on pages 60 and 76.

GROVER, S.; BASU, S. Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic. In: **Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education**. New York, NY, USA: ACM, 2017. (SIGCSE '17), p. 267–272. ISBN 978-1-4503-4698-6. Available: <http://doi.acm.org/10.1145/3017680.3017723>. Citations on pages 24, 26, and 135.

HAIR, J. F.; C., B. W.; BABIN, B.; ANDERSON, R. E. **Multivariate Data Analysis**. 7. ed. [S.l.]: Pearson Prentice Hall, 2010. Citations on pages 114, 118, and 119.

HAMILTON, M.; WEISS, M. **The Power of Storytelling in the Classroom**. Richard C. Owen Publishers, Inc, 2005. Access in: 2021-05-01. Available: <https://www.rcowen.com/PDFs/CTS%20Ch%201%20for%20website.pdf>. Citations on pages 47, 48, 96, and 99.

HERMANS, F.; AIVALOGLOU, E. Do code smells hamper novice programming? a controlled experiment on scratch programs. p. 1–10, May 2016. Citation on page 39.

HUTCHESON, G. D.; SOFRONIOU, N. **The multivariate social scientist: Introductory statistics using generalized linear models**. [S.l.]: Sage, 1999. Citation on page 118.

HWANG, G.; LI, K.; LAI, C.-L. Trends and strategies for conducting effective stem research and applications: a mobile and ubiquitous learning perspective. **International Journal of Mobile Learning and Organisation**, 2020. Citation on page 38.

INSLEY, S. Augmented reality: Merging the virtual and the real. **Oregon State University**, 2003. Citation on page 56.

JERALD, J. **The VR Book: Human-Centered Design for Virtual Reality**. New York, NY, USA: Association for Computing Machinery and Morgan & Claypool, 2016. ISBN 978-1-97000-112-9. Citations on pages 49, 50, 51, 52, 53, 55, 59, and 75.

JOHNSON, L.; ADAMS, S.; CUMMINS, M.; ESTRADA, V.; FREEMAN, A.; HALL, C. **NMC Horizon Report: 2016 Higher Education**. Austin, Texas: The New Media Consortium, 2016. Available: <http://cdn.nmc.org/media/2016-nmc-horizon-report-he-EN.pdf>. Citations on pages 23, 34, and 135.

JONASSEN, D. H. Computadores, ferramentas cognitivas - desenvolver o pensamento crítico nas escolas. **Porto: Porto Editora**, 2007. Citations on pages 38 and 49.

JORDINE, T.; LIANG, Y.; IHLER, E. A mobile-device based serious gaming approach for teaching and learning java programming. In: **2014 IEEE Frontiers in Education Conference (FIE) Proceedings**. [S.l.: s.n.], 2014. p. 1–5. ISSN 0190-5848. Citations on pages 24, 45, and 135.

KALELIOGLU, F. A new way of teaching programming skills to k-12 students: Code.org. **Computers in Human Behavior**, v. 52, p. 200 – 210, 2015. ISSN 0747-5632. Available: <http://www.sciencedirect.com/science/article/pii/S0747563215004288>. Citations on pages 23, 24, 30, and 135.

KELLEHER, C.; PAUSCH, R. Using storytelling to motivate programming. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, p. 58–64, 2007. ISSN 0001-0782. Available: <https://doi.org/10.1145/1272516.1272540>. Citation on page 49.

KIRNER, C.; SISCOUTTO, R. Realidade virtual e aumentada: conceitos, projeto e aplicações. In: **Livro do IX Symposium on Virtual and Augmented Reality, Petrópolis (RJ), Porto Alegre: SBC**. [S.l.: s.n.], 2007. Citations on pages 56, 57, and 59.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. **Engineering**, Citeseer, v. 2, n. EBSE 2007-001, 2007. Citation on page 61.

KOLLMANSBERGER, S. Helping students build a mental model of computation. In: **Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education**. New York, NY, USA: ACM, 2010. (ITiCSE '10), p. 128–131. ISBN 978-1-60558-820-9. Available: <http://doi.acm.org/10.1145/1822090.1822127>. Citation on page 24.

KUKULSKA-HULME, A.; TRAXLER, J. Design for mobile and wireless technologies. **Rethinking pedagogy for the digital age: designing and delivering e-learning**, Routledge, London, p. 180–192, 2007. Citation on page 25.

KYFONIDIS, C.; MOUMOUTZIS, N.; CHRISTODOULAKIS, S. Block-c: A block-based programming teaching tool to facilitate introductory c programming courses. p. 570–579, April 2017. ISSN 2165-9567. Citation on page 41.

LAARNI, J.; RAVAJA, N.; SAARI, T.; HARTMANN, T.; SCHRAMM, H. Ways to measure spatial presence: Review and future directions. p. 139–185, 01 2015. Citation on page 53.

LANDIS, J. R.; KOCH, G. G. The measurement of observer agreement for categorical data. **Biometrics**, Wiley International Biometric Society, 1977. ISSN 0006341X, 15410420. Available: <http://www.jstor.org/stable/2529310>. Citation on page 119.

LEAL, E. A.; ALBERTIN, A. L.; PEREIRA, J. M.; NOMELINI, Q. S. S. Utilização da análise fatorial para identificação dos fatores determinantes da aceitação do uso de tecnologias de informação na educação à distância. **Anais do Encontro da Associação Nacional de Pós-Graduação e Pesquisa em Administração, Rio de Janeiro, Brasil**, 2011. Citation on page 109.

LEE, J.; JEON, C.; ; KIM, M. A study on gamepad/gaze based input processing for mobile platform virtual reality contents. **Journal of the Korea Computer Graphics Society**, v. 22, p. 31–41, 07 2016. Citation on page 79.

LISENBEE, P.; FORD, C. Engaging students in traditional and digital storytelling to make connections between pedagogy and children's experiences. **Early Childhood Education Journal**, 2018. Citations on pages 47, 48, and 99.

LOMBARD, M.; DITTON, T. At the heart of it all: The concept of presence. **Journal of computer-mediated communication**, Oxford University Press Oxford, UK, v. 3, n. 2, p. JCMC321, 1997. Citation on page 53.

LUXTON-REILLY, A.; SIMON; ALBLUWI, I.; BECKER, B. A.; GIANNAKOS, M.; KUMAR, A. N.; OTT, L.; PATERSON, J.; SCOTT, M. J.; SHEARD, J.; SZABO, C. Introductory programming: A systematic literature review. In: **Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education**. New York, NY, USA: ACM, 2018. (ITiCSE 2018 Companion), p. 55–106. ISBN 978-1-4503-6223-8. Available: <http://doi.acm.org/10.1145/3293881.3295779>. Citations on pages 23, 26, 29, and 34.

MAGNENAT, S.; BEN-ARI, M.; KLINGER, S.; SUMNER, R. W. Enhancing Robot Programming with Visual Feedback and Augmented Reality. **Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '15**, n. Figure 1, p. 153–158, 2015. ISSN 1942647X. Available: <http://dl.acm.org/citation.cfm?doid=2729094.2742585>. Citations on pages 24 and 65.

MAKRANSKY, G.; LILLEHOLT, L. A structural equation modeling investigation of the emotional value of immersive virtual reality in education. **Educational Technology Research and Development**, Springer Link, v. 66, n. 5, p. 1141–1164, 10 2018. ISSN 1042-1629. Citation on page 73.

MANNILA, L.; DAGIENE, V.; DEMO, B.; GRGURINA, N.; MIROLO, C.; ROLANDSSON, L.; SETTLE, A. Computational thinking in k-9 education. In: **Proceedings of the Working Group Reports of the 2014 on Innovation amp; Technology in Computer Science Education Conference**. New York, NY, USA: Association for Computing Machinery, 2014. (ITiCSE-WGR '14), p. 1–29. ISBN 9781450334068. Available: <https://doi.org/10.1145/2713609.2713610>. Citations on pages 32 and 33.

MARCOLINO, A.; BARBOSA, E. Towards an m-learning requirements catalog for the development of educational applications for the teaching of programming. In: . [S.l.: s.n.], 2016. p. 1–5. Citation on page 46.

MARCOLINO, A.; BARBOSA, E. F. Softwares educacionais para o ensino de programaçao: Um mapeamento sistemático. v. 26, n. 1, p. 190, 2015. Available: <http://br-ie.org/pub/index.php/sbie/article/view/5150>. Citations on pages 24 and 38.

MARCOLINO, A. S.; BARBOSA, E. A survey on problems related to the teaching of programming in brazilian educational institutions. In: **2017 IEEE Frontiers in Education Conference (FIE)**. [S.l.: s.n.], 2017. p. 1–9. Citations on pages 26, 30, 35, 36, and 37.

MARCOLINO, A. S.; BARBOSA, E. F. Towards a software product line architecture to build m-learning applications for the teaching of programming. In: **Proceedings of the 50th Hawaii International Conference on System Sciences**. [S.l.: s.n.], 2017. Citation on page 47.

MASSO, N.; GRACE, L. Shapemaker: A game-based introduction to programming. **Proceedings of CGAMES'2011 USA - 16th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational and Serious Games**, p. 168–171, 2011. ISSN 14337347. Citations on pages 23, 29, and 65.

MEERBAUM-SALANT, O.; ARMONI, M.; BEN-ARI, M. M. Learning computer science concepts with scratch. In: **Proceedings of the Sixth International Workshop on Computing Education Research**. New York, NY, USA: ACM, 2010. (ICER '10), p. 69–76. ISBN 978-1-4503-0257-9. Available: <http://doi.acm.org/10.1145/1839594.1839607>. Citation on page 38.

MESÍA, N. S.; SANZ, C.; GORGA, G. Augmented reality for programming teaching. Student satisfaction analysis. **Proceedings - 2016 International Conference on Collaboration Technologies and Systems, CTS 2016**, p. 165–171, 2016. Citations on pages 65 and 68.

MILGRAM, P.; TAKEMURA, H.; UTSUMI, A.; KISHINO, F. Augmented reality: a class of displays on the reality-virtuality continuum. v. 2351, p. 282–292, 1994. ISSN 0277786X. Available: <http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=981543>. Citations on pages 51, 56, 57, 58, and 65.

MINELLI, R.; MOCCI, A.; LANZA, M. I know what you did last summer: An investigation of how developers spend their time. In: **Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension**. Piscataway, NJ, USA: IEEE Press, 2015. (ICPC '15), p. 25–35. Available: <http://dl.acm.org/citation.cfm?id=2820282.2820289>. Citation on page 24.

MOHAMAD, S. N. H.; PATEL, A.; LATIH, R.; QASSIM, Q.; TEW, Y. Block-based programming approach: challenges and benefits. p. 1–5, July 2011. ISSN 2155-6830. Citations on pages 39 and 98.

MOSKAL, B.; LURIE, D.; COOPER, S. Evaluating the effectiveness of a new instructional approach. In: **Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education**. New York, NY, USA: ACM, 2004. (SIGCSE '04), p. 75–79. ISBN 1-58113-798-2. Available: <http://doi.acm.org/10.1145/971300.971328>. Citation on page 42.

MOTA, J. M.; RUIZ-RUBE, I.; DODERO, J. M.; ARNEDILLO-SÁNCHEZ, I. Augmented reality mobile app development for all. **Computers and Electrical Engineering**, v. 65, p. 250–260, 2018. ISSN 00457906. Citations on pages 43 and 65.

NESTEL, D.; NG, A.; GRAY, K.; HILL, R.; VILLANUEVA, E.; KOTSANAS, G.; OATEN, A.; BROWNE, C. Evaluation of mobile learning: Students' experiences in a new rural-based medical school. **BMC medical education**, v. 10, p. 57, 08 2010. Citation on page 47.

NIELSEN, J.; MOLICH, R. Heuristic evaluation of user interfaces. In: **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. Association for Computing Machinery, 1990. (CHI '90). ISBN 0201509326. Available: <https://doi.org/10.1145/97243.97281>. Citations on pages 124, 126, 130, and 133.

NOURI, J.; ZHANG, L.; MANNILA, L.; NORéN, E. Development of computational thinking, digital competence and 21st century skills when learning programming in k-9. **Education Inquiry**, Routledge, v. 11, n. 1, p. 1–17, 2020. Available: <https://doi.org/10.1080/20004508.2019.1627844>. Citations on pages 24, 36, and 135.

NUNES, F. L.; MACHADO, L. S.; MORAES, R. M. Evolution of virtual and augmented reality in health: A reflection from 15 years of svr. In: **2014 XVI Symposium on Virtual and Augmented Reality**. [S.l.: s.n.], 2014. Citation on page 133.

OCULUS. **Oculus Go**. 2019. Access in: 2019-02-28. Available: <https://www.oculus.com/go/>. Citation on page 59.

____. **Oculus Rift**. 2019. Access in: 2019-02-28. Available: <https://www.oculus.com/rift/>. Citation on page 59.

OH, H.; DESHMANE, A.; LI, F.; HAN, J. Y.; STEWART, M.; TSAI, M.; XU, X.; OAKLEY, I. The Digital Dream Lab: Tabletop puzzle blocks for exploring programmatic concepts. **Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction - TEI '13**, p. 51, 2013. Available: <http://dl.acm.org/citation.cfm?doid=2460625.2460633>. Citations on pages 23 and 65.

O'MALLEY, C.; VAVOULA, G.; GLEW, J.; TAYLOR, J.; SHARPLES, M.; LEFRERE, P.; LONSDALE, P.; NAISMITH, L.; WAYCOTT, J. **Guidelines for learning/teaching/tutoring in a mobile environment**. 2005. Public deliverable from the MOBILearn project (D.4.1). Available: <https://hal.archives-ouvertes.fr/hal-00696244>. Citation on page 43.

ORTEGA, M.; TOLEDO, J. J.; LUNA-GARCÍA, H.; VELÁZQUEZ-ITURBIDE, J. Á.; GÓMEZ-PASTRANA, R. A.; REDONDO, M. A.; MOLINA, A. I.; BRAVO, C.; LACAVE, C.; ARROYO, Y.; SÁNCHEZ, S.; GARCÍA, M. Á.; COLLAZOS, C. A. iProg: Development of Immersive Systems for the Learning of Programming. **Proceedings of the XVIII International Conference on Human Computer Interaction - Interacción '17**, Part F1311, p. 1–6, 2017. ISSN 0006-3002. Available: <http://dl.acm.org/citation.cfm?doid=3123818.3123874>. Citation on page 65.

OZDAMLI, F.; CAVUS, N. Basic elements and characteristics of mobile learning. **Procedia - Social and Behavioral Sciences**, v. 28, p. 937 – 942, 2011. ISSN 1877-0428. World Conference on Educational Technology Researches - 2011. Available: <http://www.sciencedirect.com/science/article/pii/S1877042811026127>. Citation on page 43.

PANWONG, P.; KEMAVUTHANON, K. Problem-based learning framework for junior software developer: Empirical study for computer programming students. **Wirel. Pers. Commun.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 76, n. 3, p. 603–613, Jun. 2014. ISSN 0929-6212. Available: <http://dx.doi.org/10.1007/s11277-014-1728-9>. Citation on page 35.

PAPERT, S. **Mindstorms: Children, Computers, and Powerful Ideas**. USA: Basic Books, Inc., 1980. ISBN 0465046274. Citations on pages 23 and 30.

PEARS, A.; SEIDMAN, S.; MALMI, L.; MANNILA, L.; ADAMS, E.; BENNEDSEN, J.; DEVLIN, M.; PATERSON, J. A survey of literature on the teaching of introductory programming. **SIGCSE Bull.**, ACM, New York, NY, USA, v. 39, n. 4, p. 204–223, Dec. 2007. ISSN 0097-8418. Available: <http://doi.acm.org/10.1145/1345375.1345441>. Citation on page 37.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, v. 64, p. 1 – 18, 2015. ISSN 0950-5849. Available: <http://www.sciencedirect.com/science/article/pii/S0950584915000646>. Citation on page 61.

PETTICREW, M.; ROBERTS, H. **Systematic Reviews in the Social Sciences: A Practical Guide**. Blackwell Pub., 2006. ISBN 9781405121118. Available: <https://books.google.com.br/books?id=_Ly3aPhTkbkC>. Citation on page 61.

PFEUFFER, K.; MAYER, B.; MARDANBEGI, D.; GELLERSEN, H. Gaze + pinch interaction in virtual reality. In: **Proceedings of the 5th Symposium on Spatial User Interaction**. New York, NY, USA: Association for Computing Machinery, 2017. (SUI '17), p. 99–108. ISBN 9781450354868. Available: <https://doi.org/10.1145/3131277.3132180>. Citation on page 79.

PITEIRA, M.; COSTA, C. Learning computer programming: Study of difficulties in learning programming. In: . [S.l.: s.n.], 2013. p. 75–80. Citations on pages 26 and 35.

PITEIRA, M.; HADDAD, S. R. Innovate in your program computer class: An approach based on a serious game. In: **Proceedings of the 2011 Workshop on Open Source and Design of Communication**. New York, NY, USA: ACM, 2011. (OSDOC '11), p. 49–54. ISBN 978-1-4503-0873-1. Available: <http://doi.acm.org/10.1145/2016716.2016730>. Citations on pages 34 and 96.

POOLE, M. Design of a blocks-based environment for introductory programming in python. p. 31–34, Oct 2015. Citation on page 42.

POWELL, W.; POWELL, V.; BROWN, P.; COOK, M.; UDDIN, J. Getting around in google cardboard – exploring navigation preferences with low-cost mobile vr. In: **2016 IEEE 2nd Workshop on Everyday Virtual Reality (WEVR)**. [S.l.: s.n.], 2016. Citation on page 59.

PRICE, T. W.; BARNES, T. Comparing textual and block interfaces in a novice programming environment. In: **Proceedings of the Eleventh Annual International Conference on International Computing Education Research**. New York, NY, USA: ACM, 2015. (ICER '15), p. 91–99. ISBN 978-1-4503-3630-7. Available: <http://doi.acm.org/10.1145/2787622.2787712>. Citation on page 42.

QIAN, Y.; LEHMAN, J. Students' misconceptions and other difficulties in introductory programming: A literature review. Association for Computing Machinery, v. 18, 2017. Available: <https://doi.org/10.1145/3077618>. Citations on pages 34 and 35.

QUEIROZ, A. C. M.; TORI, R.; NASCIMENTO, A. Realidade virtual na educação: Panorama das pesquisas no brasil. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2017. p. 203. Citation on page 26.

RAABE, A. L. A.; BRACKMANN, C. P.; CAMPOS, F. R. **Currículo de referência em tecnologia e computação: da educação infantil ao ensino fundamental**. São Paulo: Centro de Inovação para a Educação Brasileira (CIEB), 2018. Ebook in pdf. Citations on pages 23, 25, 29, 31, 33, 36, 87, 90, 91, 93, 95, 96, 97, 98, 105, 107, and 136.

REEVES, T. C. Interactive learning techniques. In: ____. **Encyclopedia of the Sciences of Learning**. Boston, MA: Springer US, 2012. p. 1610–1611. ISBN 978-1-4419-1428-6. Available: <https://doi.org/10.1007/978-1-4419-1428-6_331>. Citation on page 101.

RESNICK, M.; MALONEY, J.; MONROY-HERNáNDEZ, A.; RUSK, N.; EASTMOND, E.; BRENNAN, K.; MILLNER, A.; ROSENBAUM, E.; SILVER, J.; SILVERMAN, B.; KAFAI, Y. Scratch: Programming for all. **Commun. ACM**, ACM, New York, NY, USA, v. 52, n. 11, p. 60–67, Nov. 2009. ISSN 0001-0782. Available: <http://doi.acm.org/10.1145/1592761.1592779>. Citations on pages 39, 40, and 49.

RIDEL, D.; TRIDICO, S.; BRANCO, L. H. C.; MALDONADO, J. C.; BRANCO, K. C. Technovation hackday @ icmc-usp um instrumento de difusão e articulação de meninas na computação. In: **Anais do XII Women in Information Technology**. Porto Alegre, RS, Brasil: SBC, 2018. Available: <https://sol.sbc.org.br/index.php/wit/article/view/3397>. Citation on page 33.

ROBINS, A.; ROUNTREE, J.; ROUNTREE, N. Learning and teaching programming: A review and discussion. **Computer Science Education**, Routledge, v. 13, n. 2, p. 137–172, 2003. Available: <https://doi.org/10.1076/csed.13.2.137.14200>. Citations on pages 24, 29, 30, 34, and 96.

RODELLO, I. A.; SANCHES, S. R. R.; SEMENTILLE, A. C.; BREGA, J. R. F. Realidade misturada: conceitos, ferramentas e aplicações. **Revista Brasileira de Computação Aplicada**, v. 2, n. 2, p. 2–16, 2010. Available: <http://seer.upf.br/index.php/rbca/article/view/941>. Citations on pages 50, 53, 56, 57, 58, 59, and 60.

ROQUE, R. V. **OpenBlocks: an extendable framework for graphical block programming systems**. Phd Thesis (PhD Thesis) — Massachusetts Institute of Technology, MIT, 2007. Master's thesis. Citation on page 43.

RUBENS, N.; KAPLAN, D.; OKAMOTO, T. E-learning 3.0: Anyone, anywhere, anytime, and ai. In: CHIU, D. K. W.; WANG, M.; POPESCU, E.; LI, Q.; LAU, R. (Ed.). **New Horizons in Web Based Learning**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 171–180. ISBN 978-3-662-43454-3. Citation on page 32.

SALAR, R.; ARICI, F.; CALIKLAR, S.; YILMAZ, R. A model for augmented reality immersion experiences of university students studying in science education. **Journal of Science Education and Technology**, v. 29, 01 2020. Citation on page 73.

SANTANA, B.; ARAUJO, L.; BITTENCOURT, R. Computação e eu: Uma proposta de educação em computação para o sexto ano do ensino fundamental ii. In: . [S.l.: s.n.], 2019. Citations on pages 35 and 36.

SANTOS, S.; TEDESCO, P.; BORBA, M.; BRITO, M. Innovative approaches in teaching programming: A systematic literature review. In: **Proceedings of the 12th International Conference on Computer Supported Education - Volume 1: CSEDU,**. [S.l.]: SciTePress, 2020. ISBN 978-989-758-417-6. Citation on page 37.

SBC. **Diretrizes para Ensino de Computação na Educação Básica**. 2019. Access in: 2021-01-10. Available: <https://www.sbc.org.br/educacao/diretrizes-para-ensino-de-computacao-na-educacao-basica>. Citations on pages 25, 32, 36, 90, 91, 93, 95, 97, 98, 104, and 107.

SBC, S. B. de C. **Diretrizes para Ensino de Computação na Educação Básica**. 2019. Access in: 2021-01-10. Available: <https://www.sbc.org.br/educacao/diretrizes-para-ensino-de-computacao-na-educacao-basica>. Citations on pages 23, 24, 29, 32, 74, 85, 87, 88, 90, 96, 97, 136, and 137.

SEGURA, R. J.; PINO, F. J. del; OGáYAR, C. J.; RUEDA, A. J. Vr-ocks: A virtual reality game for learning the basic concepts of programming. **Computer Applications in Engineering Education**, 2020. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.22172>. Citations on pages 26, 96, 99, 108, 135, and 136.

SENTANCE, S.; CSIZMADIA, A. Computing in the curriculum: Challenges and strategies from a teacher's perspective. **Education and Information Technologies**, v. 22, 03 2017. Citations on pages 24, 35, and 37.

SERALIDOU, E.; DOULIGERIS, C. Learning programming by creating games through the use of structured activities in secondary education in greece. **Education and Information Technologies**, 2021. Citations on pages 35 and 38.

SHARMA, S.; OSSUETTA, E. Virtual Reality Instructional Modules in Education Based on Gaming Metaphor. **Electronic Imaging**, v. 2017, n. 3, p. 11–18, 2017. ISSN 2470-1173. Available: <http://www.ingentaconnect.com/content/10.2352/ISSN.2470-1173.2017.3.ERVR-090>. Citation on page 65.

SHI, J.; WHITE, s. Work-in-progress: Learning to program in a connected world. In: . [S.l.: s.n.], 2013. p. 229–232. ISBN 978-1-4673-5627-5. Citation on page 25.

SILVA, J. V. de M.; RODELLO, I. A. Fatores determinantes para a elaboração de estratégias com vistas à aceitação e uso da realidade aumentada em cenários de negócio. **Universitas: Gestão e TI**, 2017. Citation on page 109.

SLATER, M. Immersion and the illusion of presence in virtual reality. **British Journal of Psychology**, v. 109, n. 3, 2018. Citation on page 53.

SLATER, M.; WILBUR, S. A framework for immersive virtual environments five: Speculations on the role of presence in virtual environments. **Presence: Teleoper. Virtual Environ.**, MIT Press, Cambridge, MA, USA, v. 6, n. 6, p. 603–616, Dec. 1997. ISSN 1054-7460. Available: <http://dx.doi.org/10.1162/pres.1997.6.6.603>. Citations on pages 52 and 53.

SLEEMAN, D. H. The challenges of teaching computer programming. **Communications of the ACM**, Association for Computing Machinery, v. 29, n. 9, 1986. ISSN 0001-0782. Available: <https://doi.org/10.1145/6592.214913>. Citations on pages 34, 35, 36, and 135.

SOUZA, D. M.; BATISTA, M. Helena da S.; BARBOSA, E. Problemas e dificuldades no ensino de programação: Um mapeamento sistemático. **Revista Brasileira de Informática na Educação**, v. 24, p. 39, 08 2016. Citations on pages 24, 26, 34, 36, and 37.

(SP), S. P. **Secretaria Municipal de Educação. Coordenadoria Pedagógica. Currículo da Cidade: Ensino Fundamental: Tecnologias para Aprendizagem**. São Paulo: SME/COPED, 2017. Access in: 2021-03-19. Available: <http://portal.sme.prefeitura.sp.gov.br/Portals/1/Files/47275.pdf>. Citation on page 33.

STIGALL, J.; SHARMA, S. Virtual reality instructional modules for introductory programming courses. **ISEC 2017 - Proceedings of the 7th IEEE Integrated STEM Education Conference**, v. 00, n. c, p. 34–42, 2017. ISSN 2330-331X. Citation on page 65.

SUTCLIFFE, A.; GAULT, B. Heuristic evaluation of virtual reality applications. **Interacting with Computers**, 2004. Citations on pages 108, 124, 126, 127, 128, 129, 130, and 131.

SUTHERLAND, I. E. The ultimate display. **Proceedings in The Congress of the International Federation of Information Processing (IFIP)**, p. 506–508, 1965. Citations on pages 50 and 56.

SYROCKA, A. **Why kids should learn programming?** 2017. Access in: 2021-03-22. Available: <https://www.robocamp.eu/en/blog/why-teach-kids-programming/>. Citation on page 34.

TANIELU, T.; 'AKAU'OLA, R.; VAROY, E.; GIACAMAN, N. Combining analogies and virtual reality for active and visual object-oriented programming. In: **Proceedings of the ACM Conference on Global Computing Education**. Association for Computing Machinery, 2019. (CompEd '19). ISBN 9781450362597. Available: <https://doi.org/10.1145/3300115.3309513>. Citations on pages 108, 135, and 136.

TENG, C. H.; CHEN, J. Y. An augmented reality environment for learning openGL programming. **Proceedings in The IEEE 9th International Conference on Ubiquitous Intelligence and Computing and IEEE 9th International Conference on Autonomic and Trusted Computing, UIC-ATC 2012**, p. 996–1001, 2012. Citation on page 65.

TILLMANN, N.; MOSKAL, M.; HALLEUX, J. de; FAHNDRICH, M.; BISHOP, J.; SAMUEL, A.; XIE, T. The future of teaching programming is on mobile devices. In: **Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education**. New York, NY, USA: ACM, 2012. (ITiCSE '12), p. 156–161. ISBN 978-1-4503-1246-2. Available: <http://doi.acm.org/10.1145/2325296.2325336>. Citations on pages 24, 26, 46, and 135.

TORI, R. **Educação sem distância: as tecnologias interativas na redução de distâncias em ensino e aprendizagem**. [S.l.]: São Paulo: Editora Senac São Paulo, 2010. Citations on pages 38, 52, and 137.

____. A presença das tecnologias interativas na edução. **Revista de Computação e Tecnologia (ReCeT). ISSN 2176-7998**, v. 2, n. 1, p. 4–16, 2010. Citation on page 25.

TORI, R.; HOUNSELL, M. d. S. **Introdução a realidade virtual e aumentada**. Porto Alegre (RS): Editora SBC, 2018. 940 p. ISBN 978-85-7669-446-5. Citations on pages 25, 26, 38, 47, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 107, 108, 122, 124, 126, and 133.

TORI, R.; KIRNER, C.; SISCOUTTO, R. **Fundamentos e Tecnologia de Realidade Virtual e Aumentada**. [S.l.: s.n.], 2006. 422 p. ISBN 8576690683. Citations on pages 50 and 59.

TRAXLER, J. Learning in a mobile age. **International Journal of Mobile and Blended Learning (IJMBL)**, IGI Global, v. 1, n. 1, p. 1–12, 2009. Citation on page 25.

TRIDICO, S.; RIDEL, D.; FIORAVANTI, M. L.; BRANCO, L.; MALDONADO, J.; GUESSI, M.; BRANCO, K. C. Ações para a inclusão feminina na era digital: Despertando o interesse em programação. In: . [S.l.: s.n.], 2018. Citation on page 33.

UNESCO. Skills for a connected world: Report of the UNESCO Mobile Learning Week 2018. n. March, p. 6, 2018. Citations on pages 23 and 135.

VALENTE, J. A. Integração do pensamento computacional no currículo da educação básica: Diferentes estratégias usadas e questões de formação de professores e avaliação do aluno. **Revista e-Curriculum**, v. 14, n. 3, p. 864–897, 2016. ISSN 1809-3876. Available: <https://revistas.pucsp.br/index.php/curriculum/article/view/29051>. Citations on pages 23, 29, 31, 32, 33, and 34.

VENKATESH, V.; MORRIS, M.; DAVIS, G.; DAVIS, F. User acceptance of information technology: Toward a unified view. **MIS Quarterly**, 2003. Citations on pages 108, 109, 113, 116, 121, and 123.

VINAYAGAMOORTHY, V.; GARAU, M.; STEED, A.; SLATER, M. An eye gaze model for dyadic interaction in an immersive virtual environment: Practice and experience. **Comput. Graph. Forum**, v. 23, p. 1–12, 01 2004. Citation on page 79.

VINAYAKUMAR, R.; SOMAN, K.; MENON, P. Digital storytelling using scratch: Engaging children towards digital storytelling. In: **2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)**. [S.l.: s.n.], 2018. Citation on page 47.

VINCUR, J.; KONOPKA, M.; TVAROZEK, J.; HOANG, M.; NAVRAT, P. Cubely: Virtual Reality Block-Based Programming Environment. **Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology**, n. 2, p. 84, 2017. Citations on pages 24, 25, 26, 43, 65, 96, 99, 108, 135, and 136.

VIVE, H. **Vive PRO**. 2019. Access in: 2019-02-28. Available: <https://www.vive.com/us/product/vive-pro/>. Citation on page 59.

VOSINAKIS, S.; KOUTSABASIS, P.; ANASTASSAKIS, G. A platform for teaching logic programming using virtual worlds. **Proceedings - IEEE 14th International Conference on Advanced Learning Technologies, ICALT 2014**, p. 657–661, 2014. ISSN 2161-3761. Citations on pages 50 and 65.

VRACHNOS, E.; JIMOYIANNIS, A. Design and evaluation of a web-based dynamic algorithm visualization environment for novices. **Procedia Computer Science**, v. 27, p. 229 – 239, 2014. ISSN 1877-0509. 5th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion, DSAI 2013. Available: <http://www.sciencedirect.com/science/article/pii/S1877050914000283>. Citation on page 24.

WEINTROP, D. Block-based programming in computer science education. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 62, n. 8, p. 22–25, Jul. 2019. ISSN 0001-0782. Available: <https://doi.org/10.1145/3341221>. Citations on pages 87 and 135.

WEINTROP, D.; SHEPHERD, D. C.; FRANCIS, P.; FRANKLIN, D. Blockly goes to work: Block-based programming for industrial robots. p. 29–36, Oct 2017. Citations on pages 39, 40, and 42.

WEINTROP, D.; WILENSKY, U. To block or not to block, that is the question: Students' perceptions of blocks-based programming. In: **Proceedings of the 14th International Conference on Interaction Design and Children**. New York, NY, USA: ACM, 2015. (IDC '15), p. 199–208. ISBN 978-1-4503-3590-4. Available: <http://doi.acm.org/10.1145/2771839.2771860>. Citations on pages 39, 43, 96, and 98.

WEXLER, S.; BROWN, J.; METCALF, D.; ROGERS, D.; WAGNER, E. Mobile learning: What it is, why it matters, and how to incorporate it into your learning strategy. **Guild Research**, 2008. Citations on pages 25, 43, and 47.

Wikimedia Commons. **File:6DOF.svg**. 2020. Access in: 2020-7-12. Available: <https://commons.wikimedia.org/wiki/File:6DOF.svg>. Citation on page 75.

WING, J. M. Computational thinking: What and why? In: . The Link. The magazine of Carnegie Mellon University's School of Computer Science, 2011. Access in: 2021-03-07. Available: <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>. Citations on pages 29 and 74.

WOLBER, D.; ABELSON, H.; SPERTUS, E.; LOONEY, L. **App Inventor - Create Your Own Android Apps.** [S.l.]: O'Reilly Media, Inc., 2011. ISBN 978-1-449-39748-7. Citations on pages 39, 40, 41, and 43.

WRIGHT, C.; BACIGALUPA, C.; BLACK, T.; BURTON, M. Windows into children's thinking: A guide to storytelling and dramatization. **Early Childhood Education Journal**, v. 35, p. 363–369, 2008. Citations on pages 48 and 99.

ZAMFIRACHE, V.; OLTEANU, A.; TAPUS, N. Collaborative learning assistant for android. p. 1–6, Jan 2013. ISSN 2068-1038. Citation on page 25.

ZHAI, S.; MORIMOTO, C.; IHDE, S. Manual and gaze input cascaded (magic) pointing. In: **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 1999. (CHI '99), p. 246–253. ISBN 0201485591. Available: <https://doi.org/10.1145/302979.303053>. Citation on page 79.

ZHANG, H.; BABAR, M. A.; TELL, P. Identifying relevant studies in software engineering. **Information and Software Technology**, 2011. ISSN 0950-5849. Available: <http://www.sciencedirect.com/science/article/pii/S0950584910002260>. Citation on page 62.


ICMC
USP
SÃO CARLOS