FEDERAL UNIVERSITY OF CEARÁ

CENTER OF SCIENCE

DEPARTMENT OF COMPUTER

POST-GRADUATION PROGRAM IN COMPUTER SCIENCE

MASTER DEGREE IN COMPUTER SCIENCE

JOSÉ MARTÔNIO LOPES DE MORAES JÚNIOR

ENHANCING ORIENTATION AND MOBILITY SKILLS FOR PERSONS WITH VISUAL IMPAIRMENTS THROUGH VR GAMES: INTRODUCING THE EDKIT UNITY PACKAGE

FORTALEZA

2024

JOSÉ MARTÔNIO LOPES DE MORAES JÚNIOR

ENHANCING ORIENTATION AND MOBILITY SKILLS FOR PERSONS WITH VISUAL IMPAIRMENTS THROUGH VR GAMES: INTRODUCING THE EDKIT UNITY PACKAGE

Dissertation submitted to the Post-Graduation Program in Computer Science of the Center of Science of the Federal University of Ceará, as a partial requirement for obtaining the title of Master in Computer Science. Concentration Area: Software Engineering.

Advisor: Prof. Dr. Windson Viana de Carvalho.
Co-advisor: Prof. Dr. Agebson Rocha Façanha.

FORTALEZA

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M821e Moraes Júnior, José Martônio Lopes de.
Enhancing orientation and mobility skills for persons with visual impairments through VR games : Introducing the EdKit Unity package / José Martônio Lopes de Moraes Júnior. – 2024.
148 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2024.
Orientação: Prof. Dr. Windson Viana de Carvalho.
Coorientação: Prof. Dr. Agebson Rocha Façanha.

1. Orientation and mobility. 2. Virtual reality. 3. Gestures. 4. Multisensory feedback. I. Título.

CDD 005

JOSÉ MARTÔNIO LOPES DE MORAES JÚNIOR

ENHANCING ORIENTATION AND MOBILITY SKILLS FOR PERSONS WITH VISUAL IMPAIRMENTS THROUGH VR GAMES: INTRODUCING THE EDKIT UNITY PACKAGE

Dissertation submitted to the Post-Graduation Program in Computer Science of the Center of Science of the Federal University of Ceará, as a partial requirement for obtaining the title of Master in Computer Science. Concentration Area: Software Engineering.

Approved on: November 29, 2024.

EXAMINATION BOARD

Prof. Dr. Windson Viana de Carvalho (Advisor)
Federal University of Ceará (UFC)

Prof. Dr. Agebson Rocha Façanha (Co-advisor)
Federal Institute of Education, Science and Technology of Ceará (IFCE)

Prof. Dr. Fernando Antonio Mota Trinta
Federal University of Ceará (UFC)

Prof. Dr. Joel André Ferreira dos Santos
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)

To my parents, for everything they did for me.

ACKNOWLEDGEMENTS

I thank God, for giving me strength to keep going until the end and help me overcome the most challenging times.

I thank my parents, for supporting and backing me so I could dedicate myself to this Masters' Degree from start to finish.

I thank Windson, my advisor, for all the patience and support through these three years until we were able to reach this research I'm presenting today.

I thank Agebson, my co-advisor, for all the tips and suggestions about Orientation and Mobility that helped me understand a subject that was completely new to me.

I thank Bruno, for helping me find the focus for my research, being part of the process to make it doable in time.

I thank my research team (Mariana, Phyllipe, Erivan) and "Trupe do Vento" for all the help they gave me and all the good moments through my time in this Masters' degree.

I thank all the professors that participated in my examination boards (Fernando Trinta, Ticianne Darin, Joel dos Santos) for all the critiques, recommendations and suggestions that helped improve this work up to it's current state.

Lastly, I thank everyone else that was part of this journey and I wasn't able to mention by name: my friends, family members, teachers, colleagues and many others. Nothing but my utmost gratitude!

This study was financed in part by the Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP)

“One of the joys of being a programmer is deciding just how to go about telling our clever lies and how to best express the things that happen in an imaginary world” (Takuhiro Dohta - “Breaking Conventions with The Legend of Zelda: Breath of the Wild” - Game Developers Conference 2017)

ABSTRACT

The integration of digital technologies, particularly Virtual Reality (VR) games, into Orientation and Mobility (OM) training, has shown significant potential to improve the rehabilitation process for People with Visual Impairments (PVI). These technologies offer immersive and interactive experiences that can foster greater autonomy and independence. However, VR games specifically designed for OM training are currently limited, and tools or code libraries to support their development are also scarce. To address this gap, this work presents 'EdKit', a custom Unity package that combines gestures, multimodal feedback and in-game analytics to help integrate Left-Right Discrimination concepts into VR games for PVI. EdKit also supports OM analytics, which can be used to enhance the educational experience for both students and instructors. This package was also used to create three minigames in order to fill the gaps in its feature set. Overall, developers had a positive reception for EdKit, agreeing that it is a package that makes implementation easier and more legible. However, the evaluation also revealed that EdKit is a package with a challenging initial learning curve. Participants suggested that this could be mitigated by providing more code examples, learning resources, and implementing designer-friendly workflows.

Keywords: orientation and mobility; virtual reality; VR; gestures; multisensory feedback.

RESUMO

A integração de tecnologias digitais, especialmente jogos de Realidade Virtual (VR), para o treinamento de Orientação e Mobilidade (OM), demonstra potencial significante em melhorar o processo de reabilitação para Pessoas com Deficiência Visual (PDV). Estas tecnologias ofecerem experiências imersivas e interativas que fomentam maior autonomia e independência. No entanto, jogos RV especificamente projetados para o treinamento OM são limitados hoje em dia, com ferramentas e bibliotecas de código para suporte ao desenvolvimento de tais soluções são escassos. Para resolver esta lacuna, este trabalho apresenta "EdKit", um pacote Unity personalizado que combina gestos, *feedback* multimodal e métricas para jogos para ajudar na integração de conceitos de Distinção Direita-Esquerda em jogos RV para PDVs. EdKit também suporta métricas para OM, que pode ser usado para incrementar a experiência educacional para estudantes e instrutores. Este framework foi utilizado para desenvolver três mini-jogos para suprir omissões no conjunto de funcionalidades. No geral, desenvolvedores tiveram uma recepção positiva à EdKit, concordando que o mesmo é um pacote poderoso que facilita a implementação e a torna mais legível. No entanto, a avaliação também demonstra que EdKit é um pacote avançado com uma curva inicial de aprendizado alta. Participantes sugeriram que a curva pode ser mitigada através de mais exemplos de códigos, recursos de ensino e implementando fluxos de aprendizado amigáveis à designers.

Palavras-chave: orientação e mobilidade; realidade virtual; RV; feedback multissensorial; gestos.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

1.1 Contextualization

According to “World Report on Vision”, a publication from the World Health Organization (WHO), there are 2.2 billion people worldwide with some kind of visual impairment (ORGANIZATION *et al.*, 2019). People with Visual Impairments (PVI) have one or more vision functions affected that could limit their interaction with the world and day-to-day activities. PVI could have difficulties determining their location in the environment they’re in, as well as the spatial relations between themselves and people, objects, and other reference points (LAHAV, 2022). Orientation and Mobility (OM) is a training field that focuses on enabling PVI to navigate their environment safely and independently, supporting cognitive, motor and emotional development (ORGANIZATION *et al.*, 2019; KREIMEIER; GÖTZELMANN, 2020), and is generally recommended for people who are blind or have low vision.

Developing OM concepts with PVI enables them to use other senses and residual vision to train their spatial perception skills (LOOMIS *et al.*, 2013), analyzing tactile and audio cues for tasks such as navigation, route planning, and localization (THEVIN *et al.*, 2020). By training OM skills, we allow people to understand and interact with the world with more accuracy and confidence, as well as build a mental representation of the spaces they’re in or will navigate on it (ANDRADE *et al.*, 2021).

One such concept is Left-Right Discrimination (LRD), which is the propensity that humans have to define a preference to use one side of the body more than the other through the development of neural, psychological, and motor skills (LUSSAC, 2022). Some activities involve the identification of body parts, learning to use the limbs as reference points, understanding how to use their smaller body parts such as fingers, and executing crossed commands (e.g. “Put your right hand on your left shoulder.”). Other example is discerning left and right using their body as a reference point (egocentric) or projecting their orientation system into external references (allocentric) (WATEMBERG *et al.*, 1986; LUSSAC, 2022; HOOGSTEEN *et al.*, 2022; GONÇALVES *et al.*, 2023a).

Assistive technologies make it possible to improve spatial cognition through artifacts designed for activities that develop more than one spatial skill at once. Combined with technologies such as Virtual Reality (VR), it’s possible to create engaging experiences that encourage the practice of OM exercises in an immersive and playful environment that is fully controllable. VR

gaming can effectively substitute real-world games for cognitive tasks within a limited spatial environment, yielding comparable performance and user experience metrics in both settings (CHOUEIB; BERKMAN, 2023). It allows the player to explore new situations without the dangers present in the real world. Researchers have shown OM skills acquired can be transferred between virtual and real environments (THEVIN *et al.*, 2020; LAHAV *et al.*, 2012; FAÇANHA *et al.*, 2020).

1.2 Motivation

Testing solutions directly with the target audience is crucial, especially when addressing the diverse set of needs that PVI have. These users often have unique requirements that may necessitate specific attention, like visual elements for those with low vision. To meet these needs effectively, many solutions focus on Text-to-Speech (TTS), multi-modal feedback, sonification, and audio techniques. Engaging with experts and PVI during the development process ensures these solutions create comprehensive and inclusive experiences. While recruiting PVI for participation can be challenging, evaluating individuals without visual impairments could compromise the validity of the process. Therefore, it's essential to include a diverse and representative sample of PVI and consider the full spectrum of visual impairments.

In VR, players can enhance their skills over time, leveraging their previous experiences to improve their performance in games. As interest in this emerging entertainment medium grows, it's important to refine aspects such as efficiency, immersion, information delivery, multi-sensory feedback, and autonomy to better meet user needs based on their context.

As such, integrating VR games for PVI in an OM practice model allows for more immersive experiences in virtual environments, provided the necessary audio and haptic cues are in place and the design combines the interactions that are exclusive to VR with existing PVI skills (RIBEIRO *et al.*, 2024). When combined with concepts of game analytics, they offer a new tool for OM instructors to analyze the student's performance while playing the game: they could plan future learning strategies, either in a physical or remote practice model, with the latter being an opportunity space for new research interventions (DOVE *et al.*, 2022). It is crucial to create and make available libraries and frameworks that aid developers in creating accessible VR games for PVI.

1.3 Research Goal and Hypothesis

This research presents the concept of VR games for PVI for LRD training, with the main research question defined as follows:

Main Research Question

How can we bring Left-Right Discrimination concepts for the development of Virtual Reality games for People with Visual Impairments?

Therefore, the research goal is described as follows:

Research Goal

Simplify the integration of LRD concepts in VR games for PVI.

To do so, we have the specific goals of defining the main features required for developing a game using LRD concepts, propose and evaluate an framework for a game engine that consolidates the approach, create one or more games as proof-of-concepts using this newly created artifact and evaluate the artifact's usability with VR game developers.

The present dissertation uses the following hypothesis as a guide:

Research Hypothesis

A dedicated artifact containing domain knowledge of LRD concepts makes it easier for developers to create VR games for PVI working with these concepts.

1.4 Roadmap

This dissertation is divided into six more chapters, in which we discuss the underlying concepts, present the solution artifact created for this research and discuss and answer the main research question.

Chapter 2 presents a theoretical groundwork for the fields of OM, LRD, VR, frameworks and Unity.

Chapter 2 presents the systematic mapping performed, related works identified in the process and it's results.

Chapter 4 shows the methodology used for developing and evaluating the solution artifact.

Chapter 5 defines our solution artifact known as EdKit, presenting it's design approach and main modules.

Chapter 6 details the development process of the package, highlighting the instances created in order to refine the package.

Chapter 7 presents the evaluation process done with game developers to analyze the package's usability and the results gathered from the evaluation process, discussing EdKit's advantages, disadvantages and key takeaways.

Chapter 8 concludes by presenting the contributions made by this research and possible avenues for further development.

2 BACKGROUND

This chapter presents fundamental concepts for this research, serving as a base for the dissertation. Section 2.1 focus on the Background of this work, talking about Orientation and Mobility, Left-Right Discrimination, Virtual Reality, Frameworks and Unity. Section 3 presents the State of the Art and Related Work about Virtual Reality games for People with Visual Impairments training Left-Right Discrimination concepts.

2.1 Background

2.1.1 Orientation and Mobility

OM is defined by SIMÕES and CAVACO (SIMÕES; CAVACO, 2014) as the "ability to move independently, safely and efficiently from one place to another", developing necessary concepts to live a more active life. Orientation involves using remaining senses to understand one's position and relationship to key elements in the environment. Mobility, on the other hand, refers to the ability to move through an environment safely, efficiently, and comfortably using these senses (FAÇANHA *et al.*, 2020). Therefore, it is crucial for PVI to not only read and follow routes but also learn to navigate from one location to another. They should be alerted and guided about their destination, which helps them, even if unconsciously, to build a mental map of their surroundings (MENDONÇA *et al.*, 2008).

Thus, OM is a specialized field within education dedicated to rehabilitating PVI, whether those impairments are congenital or acquired. The primary goal of OM is to equip these individuals with techniques for spatial understanding, self-protection, and navigation, fostering greater confidence and independence in their daily lives (FAÇANHA *et al.*, 2020).

Rehabilitation in this context is not a fixed process with a fixed timeline, but rather a change of attitude and a constant process for adapting to life without vision (RIBEIRO, 2013). As described by KREIMEIER; GÖTZELMANN (KREIMEIER; GÖTZELMANN, 2020), OM practices focus on developing various aspects for PVI across several domains:

- **Cognitive**: Enhancing the understanding and assimilation of concepts, the nature and function of objects, problem-solving abilities, abstraction, and the retention and transfer of skills.
- **Psychomotor**: Providing experiences to develop perception for basic and fundamental

movements, physical capabilities, motor skills, and non-verbal communication.

- **Emotional**: Supporting improvements in self-confidence, self-esteem, motivation, values, and self-image.
- **Remaining Senses**: Stimulating the effective use of residual vision in low vision scenarios, recognizing clues, defining previously identified reference points, and connecting the action space with surrounding objects.

One example of how OM enhances mobility is by helping individuals determine their positioning relative to their environment, including people, objects, and other reference points, as well as identifying shapes and potential walking routes (INDIA *et al.*, 2021). Information about the environment is crucial for PVI to navigate effectively and develop a mental representation that aids in spatial orientation, allowing them to project the navigated space and its key features (LAHAV, 2022). These models can take the form of mental maps based on Points of Interest (PoI), routes, or knowledge acquired through exploration (ANDRADE *et al.*, 2021).

At a conceptual level, OM tasks facilitate the development of strategies for creating cognitive maps that enable efficient spatial navigation (LAHAV *et al.*, 2012). By using a cognitive map, individuals can better understand spatial relationships between elements in their environment. This approach shifts the focus of exploration from continuously acquiring new information to leveraging previously identified connections (SIMÕES; CAVACO, 2014; BURTE; MONTELLO, 2017).

One approach to building cognitive maps involves two key methods:

- Perimeter Recognition: As PVI walk and scan their environment, they identify and establish the boundaries of the explored location.
- Route Creation: This method involves using linear recognition of spatial features to determine routes and use them to navigate through the environment (WILLIAMS *et al.*, 2014).

Both techniques help PVI develop a mental representation of their surroundings, aiding in navigation and spatial awareness.

One of the fundamental components used by PVI for spatial mapping is the concept of nodes, which are key points of interest within their environment. During exploration, PVI often encounter decision points—moments when they must choose a path to reach a specific goal. These decision points can occur at nodes or along the connections between nodes. Points of Interest (PoI) are fixed objects located at specific points, which are typically integrated into

routes to aid in orientation and localization (AZIZ *et al.*, 2022). Routes, on the other hand, represent a step-by-step process designed to guide a person from point A to point B (KITCHIN; JACOBSON, 1997).

When discussing navigation guidance, it's crucial to recognize that PVI have different information needs compared to those without visual impairments. While PoI are generally helpful for orientation, deviating from a route into unfamiliar areas can sometimes lead to disorientation and confusion for PVI (THINUS-BLANC; GAUNET, 1997).

To mitigate this risk, surveying the environment and creating spatial partitions can enhance navigation by providing a more comprehensive cognitive map. This approach makes the navigation process easier and more manageable (ANDRADE *et al.*, 2021).

Assistive technologies play a crucial role in developing OM skills, utilizing both digital applications (SIMÕES; CAVACO, 2014) and physical tools like tactile maps (BRULE *et al.*, 2016). By integrating OM with digital assistive technologies, we can create engaging experiences that encourage users to practice spatial orientation tasks in a playful and interactive manner. This can be achieved through virtual or hybrid environments, which offer immersive ways for users to interact and refine their navigation skills.

In OM, there are two primary approaches to providing information to users:

- **Real-Time:** This approach delivers information about the surroundings during navigation. Examples include using a guide, cane, or guiding dog as navigation aids, or employing Electronic Travel Guide (ETG) that incorporate technologies such as echolocation, GPS, sensors, and cameras to provide audio and tactile feedback.
- **Beforehand:** This method provides information about a location before the user navigates it. Examples include audio virtual environments, tactile maps, verbal descriptions, physical models, or touch screens (LAHAV *et al.*, 2012).

In this context, spatial orientation is a critical concept for PVI, involving the use of residual vision and other senses to enhance their spatial perception skills, analyzing audio and tactile cues to perform spatial tasks such as navigation and locating objects (THEVIN *et al.*, 2020). It is important to recognize that the mental spatial representations developed by PVI are not reliant on vision: instead, they are formed through alternative senses or even through the use of language (LOOMIS *et al.*, 2013).

Overall, spatial orientation is essential for PVI to develop spatial competence, enabling them to understand and interact with their surroundings with greater accuracy and confi-

dence.

Spatial cognition refers to the ability to create, retain, retrieve and utilize well-structured spatial models (CHANG *et al.*, 2019). This pervasive skill is applicable at any moment during interactions between an individual and their environment.

Spatial and temporal representations are often developed independently of vision, with the integration of said elements facilitating quicker and more conscious perception (THEVIN *et al.*, 2020; KOKKINARA *et al.*, 2015). It is also important to note that spatial demonstratives (such as the relative positioning of objects) are generally more effective for orientation and navigation than temporal demonstratives (GRIFFITHS *et al.*, 2019).

Ideomotor theory posits that action, perception, and cognition are intrinsically interconnected. Concepts related to these domains actively contribute to the development of spatial orientation, rather than merely serving as passive receptors (CHANG *et al.*, 2019).

Therefore, promoting OM is crucial for PVI due to its psychological, physical, social, and economic benefits. Most importantly, OM supports their fundamental right to navigate and participate in society on equal terms with other citizens (MACIEL, 2003).

2.1.2 Left-Right Discrimination

This work defines Left-Right Discrimination (LRD) as "the propension that humans have of preferring one side of the body over another through neuropsychomotor skills" (LUSSAC, 2022). LRD is a concept that develops over the course of a person's life: it begins with an understanding of the body's major parts, progresses to using limbs as reference points, involves the assimilation of the use of smaller body parts (e.g., fingers), and includes the ability to identify whether an object is to the left or right (WATEMBERG *et al.*, 1986).

By the age of 12, children typically have similar abilities as adults to distinguish between the left and right sides of their own body and those of others in front of them (WATEMBERG *et al.*, 1986). They're also capable of executing crossed commands, such as placing their right hand on their left shoulder, while maintaining awareness of the context (WATEMBERG *et al.*, 1986). This ability allows them to project their orientation system onto other references, thereby establishing spatial relationships more effectively (LUSSAC, 2022).

As described by OFTE (OFTE, 2002), LRD involves cognitive skills, integrating sensory information, using receptive and expressive language, taking different perspectives, and processing visual and spatial information to perform spatial orientation and motor skills, such as:

- **Plane Slicing:** Discerning between different body planes (e.g. left-right or above-below).
- **Body Image:** Awareness of one's own body, which includes identifying and locating their body parts.
- **Aligned Commands:** Commands that follow the same orientation as the person's reference frame (e.g., placing your left hand on your left shoulder).
- **Crossed Commands:** Commands that require correlating two different directions within a single action (e.g. stepping with your left foot to turn right).
- **Depth Perception:** Perceiving the distance of an object from a spatial reference point.
- **Clue Recognition:** The ability of associating an object, environment or actor with other characteristics to make it more easily identifiable (e.g. sound, shape, topography, temperature, texture)
  - Concrete: Identify specific properties of an object.
  - Functional: Identify what the object does and what it can be used for.
  - Abstract: Synthesize the object's main properties.
- **Motor Coordination:** Coordinating limbs effectively to interact with tasks or objects.
- **Dominant Side:** The preference for using one side of the body over the other for specific tasks (e.g., being more proficient with the left hand to hold a cup).
- **Spatial References:** The use of a spatial reference in the environment to perform tasks and navigate spaces. A spatial reference can be categorized into two types:
  - Egocentric: Based on the individual's own body (e.g. "raise your left hand").
  - Allocentric: Based on an external reference point (e.g. "turn left at the bus stop") (HOOGSTEEN *et al.*, 2022; GONÇALVES *et al.*, 2023a).

  While PVI generally perform equally well with egocentric references, they may face difficulties with allocentric tasks. This is because the ability to translate spatial information to external reference points through the visual channel is essential in the development of LRD skills (WATEMBERG *et al.*, 1986).

By understanding their own body's orientation, PVI can determine the spatial relationships between themselves and objects, distinguishing whether an object is to their left or right. Over time, this ability extends to recognizing spatial relationships between objects themselves.

Even for people without visual impairments, differentiating between left and right often requires a conscious effort, using body cues such as limb movements or markers to clearly define each side of the body (GORMLEY *et al.*, 2019).

Based on the perspective outlined by HEGARTY *et al.* (HEGARTY *et al.*, 2006), spatial relationships can be categorized into three scales:

- Figural: The region within the range of a person's limbs.
- Visual: The area encompassed within a person's field of view.
- Navigable: The broader environment that can be explored.

Addressing spatial tasks in relation to these body-based spatial relationships can involve two approaches(CHANG *et al.*, 2019):

- Epistemic Solving: performing actions to experiment and discover the best solution through direct experience.
- Pragmatic Solving: exploring solutions mentally before taking any physical action.

In addition to understanding spatial relationships, training plane slicing skills also enhances motor coordination between hands and feet, which is essential for executing simple tasks (e.g. walking, holding an object, pointing to a relevant item) and more complex ones (e.g. performing a crossed command). Over time, practice helps individuals develop preferences for using one side of the body for specific tasks, such as using the left hand to drink from a cup.

At the same time, factors such as emotional state and attention can significantly impact the development of LRD skills: for instance, changes in emotional state or focus can lead individuals to consciously reassess which side is their left or right, requiring them to rely on personal conscious verification mechanisms to accurately determine spatial orientations.

When focusing on LRD as a standalone topic within the OM field, we can delve deeper into its nuances and design controlled physical exercises to enhance competence, where they can serve as foundation for advancing to more complex tasks and concepts on the subject.

Mastery of these concepts involves reducing the time spent recognizing and using spatial orientations. It also requires a clear understanding of left and right, not only through basic spatial references but also within more complex scenarios. This proficiency allows individuals to efficiently apply these skills in a variety of contexts and tasks.

Beyond simply favoring one side of the body over the other, LRD also plays a crucial role in enhancing motor coordination in PVI, developing more efficient and coordinated movement patterns which can improve overall motor skills and facilitate better interaction with their environment.

2.1.3 Virtual Reality

The use of Virtual Reality (VR) solutions in scientific research enables the simulation of real-world activities in an interactive and immersive virtual environment. This technology is applied across various fields, including health (YIP; MAN, 2009) and sports (COVACI *et al.*, 2014; KARAOSMANOGLU *et al.*, 2022), to create realistic scenarios for study and practice.

Based on the studies of ELOR *et al.* (ELOR *et al.*, 2020) and SPEICHER *et al.* (SPEICHER *et al.*, 2019), VR can be categorized into several models and formats, as shown on Figure 1.

- Virtual Environments (Figure 1a): These are immersive settings that simulate complex situations using multisensory feedback to create a realistic experience (ELOR *et al.*, 2020).
- Hybrid Environments (Figure 1b): These involve physical spaces equipped with projectors and sensors that facilitate interaction with virtual elements.
- Immersive Virtual Reality (IVR) (Figure 1c): Often used interchangeably with VR, IVR refers to virtual environments that offer a deeper level of physical interaction, typically through Head-Mounted Display (HMD) and motion controllers.
- Mixed Reality (XR) (Figure 1d): Although XR has multiple definitions, SPEICHER *et al.* (SPEICHER *et al.*, 2019) describes it as "the blending of virtual and real objects in a single display, ranging from completely virtual to entirely real."

Developing VR solutions typically involves using commercial hardware such as Meta Quest, Valve Index, or HTC Vive. These devices provide the necessary head-mounted displays and motion controllers. Additionally, a game engine is employed to create and manage the virtual experiences, with popular choices including Unity, Godot, and Unreal Engine.

The emphasis on VR was selected because it leverages HMD, motion controllers, and advanced technologies such as spatial audio and hand tracking to facilitate more active physical interaction (CHANG *et al.*, 2019). Furthermore, focusing on immersion and presence within VR experiences enhances the learning process and supports the transfer of skills to everyday activities for PVI (NARASIMHAM *et al.*, 2020; CHANG *et al.*, 2019; THEVIN *et al.*, 2020; FAÇANHA *et al.*, 2020).

The use of VR by PVI allows for extensive interaction with simulated environments while maintaining full control over the experience's parameters. This approach helps avoid the risks and hazards present in real-world environments, thereby preventing accidents and unpredictable situations (THEVIN *et al.*, 2020).

Figure 1 – Types of Virtual Reality

(a) Virtual Environment (b) CAVE Hybrid Environment

(c) Immersive Virtual Reality (d) Mixed Reality

Source: (a) (REGAL *et al.*, 2018), (b) (ELOR *et al.*, 2020), (c,d) Unsplash

Beyond facilitating the transfer of skills, VR's immersion and sense of presence enable PVI to experience environments in ways that closely resemble real-world settings. To achieve this, it's crucial to implement a robust interaction framework, which may include multiple feedback mechanisms or a hierarchical model of information (HOOGSTEEN *et al.*, 2022). Nonetheless, achieving high levels of audiovisual fidelity is not always necessary, as even a lower level of fidelity can meet the immersion requirements for a virtual environment (OUMARD *et al.*, 2021; THEVIN *et al.*, 2020).

As an educational tool for OM, VR offers the significant advantage of allowing skills learned in the virtual environment to be transferred to the real world, and vice versa (LAHAV *et al.*, 2012; FAÇANHA *et al.*, 2020). Developers can leverage OM concepts to create more effective interaction mechanisms tailored to PVI.

This advantage extends to the comparison between VR gaming experiences and real-world gameplay using physical toys, as studies have shown that cognitive tasks performed

in this setting can yield performance and user experience metrics comparable to those achieved with physical toys in a confined spatial environment (CHOUEIB; BERKMAN, 2023).

It's crucial that VR experiences are designed to avoid reinforcing any stigma through their visual presentation (BRULE *et al.*, 2016). An effective physical VR experience should consider factors such as the available physical space, locomotion constraints, and the level of fidelity in the virtual environment (THEVIN *et al.*, 2020). This ensures that the experience is both inclusive and practical, accommodating the user's needs and physical limitations.

One of the challenges of using IVR is the potential for side effects from prolonged use, which can also affect PVI. Nausea is one of the most commonly reported symptoms, but various factors contribute to cybersickness following a VR session (JR, 2000), which may be linked to aspects of the experience, including the ergonomics of the devices and navigation techniques used within the virtual environment.

Solutions that simulate real-world locations, such as buildings and museums, using three-dimensional audio are employed to enhance OM skills for PVI (FAÇANHA *et al.*, 2020). Research has demonstrated that integrating auditory cues, sonar techniques, and three-dimensional audio with haptic feedback devices (e.g., joysticks and electronic canes) improves user interaction and technology acceptance, with a positive cognitive impact on OM skills (LAHAV, 2022).

Technological advancements have led to improved skills in areas such as self-location, where users identify the direction of sounds and contextualize them, and audio orientation, where they determine the origin of moving sounds and navigate accordingly.

Developers of VR games with PVI as the target audience must take advantage of common VR idioms to implement relevant features that improve their interaction experience, whether by allowing free exploration by mixing locomotion methods (HOOGSTEEN *et al.*, 2022; THEVIN *et al.*, 2020), adapting visual techniques for use by PVI (OUMARD *et al.*, 2021), maintaining realistic proportions of movement and stable representation of virtual objects (THEVIN *et al.*, 2020), or through the choice of navigation mechanisms used. The use of previously learned concepts serves to enable an experience as close to reality as possible, where PVI can use unique skills they have access to, such as echolocation and the use of routes and points of interest (ANDRADE *et al.*, 2021).

Another advantage of simulating virtual environments is the ability to control and customize the experience to enhance learning: VR allows for adjustments such as distinguishing

sounds from different directions—front, back, above, and below the user—removing visual elements to encourage reliance on other senses, and managing the proximity and diversity of sound sources to balance the auditory scene (HOOGSTEEN *et al.*, 2022; THEVIN *et al.*, 2020). These modifications help tailor the experience to better support the development of OM skills.

For game developers, it is crucial to incorporate features that facilitate interaction with VR devices for PVI. These features include screen readers, TTS systems, spatial audio, and other assistive technologies. Development typically involves using game engines such as Unity, Unreal Engine, or Godot, which provide a foundation for deploying to VR platforms.

Additionally, developers can leverage proprietary Software Development Kit (SDK) offered by HMD manufacturers. These SDKs can help tailor the experience to better meet the needs of PVI, enhancing both usability and accessibility.

In terms of commercially available VR games for PVI, *Cosmonious High*(LABS, 2023) stands out. This narrative adventure game, where players enroll in an alien high school and work to resolve various issues, includes a range of accessibility features. Although these features were added in an update post-launch, the inclusion of haptic feedback, audio descriptions, high-contrast modes, and other enhancements enables players who are legally blind or have low vision to fully engage with the game (LABS, 2023).

Overall, the use of VR by PVI demonstrates significant potential as a tool for prototyping solutions and can be employed to simulate tools and virtual environments, as well as to create engaging and empathetic experiences. This technology not only facilitates the development and testing of innovative solutions but also enriches the interaction and learning experiences for PVI.

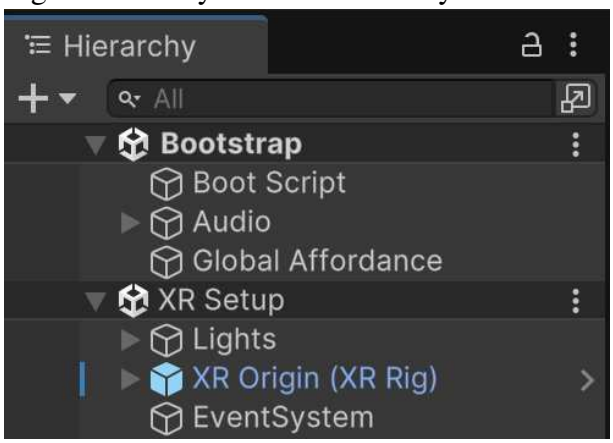
2.1.4 Unity

In order to understand the concepts that will be detailed in Chapters 5 and 6, it is important to define some basic Unity concepts and terminology in order to better define concepts around the solution artifacts.

Unity is a game engine for development of 2D and 3D games, apps and multimedia experiences that focus on cross-platform game development, being able to deploy to multiple platforms as required. The engine has VR support for multiple platforms through it's XR plugin and packages (TECHNOLOGIES, 2023).

The runtime of a Unity game is structured as a linear collection of Scenes, containing

Figure 2 – Unity's Scene Hierarchy



Source: Author

all or part of a game or application's elements, which may include characters, user interface, terrain or other types of assets (TECHNOLOGIES, 2023). The developers can choose to either have one scene loaded in memory at a time or multiple at once, loading and unloading them as necessary. Scenes are composed of game objects, which are instances representing the elements in a game. Game objects are structured as a tree hierarchy, being able to have other game objects as children of itself. Figure 2 shows an example of a scene hierarchy in Unity containing multiple scenes.

In order to define how game objects should behave, developers can add components, which are reusable and customizable blocks of functionality that can be attached to a game object in a scene. All game objects have a Transform component attached to them that cannot be removed, which stores the position, rotation and scale in the scene. You can also use components in game objects to add meshes, cameras, user interface elements, physics and more (TECHNOLOGIES, 2023).

Developers can save a configured GameObject as a Prefab, which is an asset type that works as a template, storing it and all of it's components, values and child game objects to be reused in authoring a scene or for creating instances during gameplay. Developers can also nest prefabs into other prefabs or create variants with a set of overrides while reusing the base template.

Developers can also take advantage of Scriptable Objects, which are data assets that work as reusable information containers, working as a flyweight data structure that is reusable between components and persists in memory until it's not used anymore. While each Scriptable Object created works as a reference that is instanced only once in memory, new copies can be instanced at runtime, working as new instances (TECHNOLOGIES, 2023).

To implement a new component in Unity, programmers need to create new C# scripts with a new class, inheriting from the "MonoBehaviour" class and implementing methods that compose Unity's PlayerLoop to execute code based on the stage of the lifecycle, the order of execution and respond to events such as physics callbacks, user input and more. After adding a component to a GameObject in the Editor, developers can change it's parameters, object references to other components and bind declared game events into other components.

When creating a custom type to manipulate it's parameters, it's important take into account that the Unity Editor uses Serialization to synchronize the values to be stored and reconstructed later on. The controls for manipulating the component's variables vary based on the attributes used and the variable's type: one example of this is UnityEvents, which are serialized function calls that can be configured directly on the scene without extra programming or scripting, being a useful way to create content-driven callbacks and decouple systems. Programmers also get access to tools for executing code in components across several frames, either by using Coroutines, Unity's C# Job System or C#'s native concurrency features.

In terms of specifications, the engine is build with native C/C++ internally, but all code implemented by developers is written in C# (TECHNOLOGIES, 2023). Scripting uses a limited version of C# 9.0 version, with some features not being supported such as module initializers and init-only setters (TECHNOLOGIES, 2023). Just like other C# projects, script compilation can be structured to define what gets compiled for each platform and in each assembly, as well as platform dependent compilation using preprocessor directives. A project in Unity automatically creates a assembly for runtime code and one for editor-specific code, with other packages in their project being located in their own standalone assemblies.

2.1.5 Frameworks

As new requirements and specifications arise, it is important that code is more resilient and prepared for changes: as mentioned by PICCIONI *et al.* (PICCIONI *et al.*, 2013), modern software development is built over the foundations of abstraction and modularity that allows component reuse to build new software.

A framework is defined as a "set of prefabricated software building blocks that programmers can use, extend, or customize for specific computing solutions" (BOYD, 1993). Frameworks encapsulate knowledge related to a specific domain to solve both the original problem it was created for and future solutions as if it was made for them, modeling both

dynamic behavior and structures of the problem domain through the use of patterns (BOYD, 1993; PARSONS *et al.*, 1999; JACOBSEN *et al.*, 1997).

BOYD (BOYD, 1993) also classifies frameworks into three distinct categories:

- **Application frameworks**: Encapsulates knowledge that can be reused by a wide range of applications (e.g. UI frameworks such as Unity UI)
- **Domain frameworks**: Wraps reusable expertise for a particular topic or subject (e.g. framework for training Machine Learning models such as SciPy)
- **Support frameworks**: Offers system-level services to the developer (e.g. a framework to access the device's camera)

Libraries on the other hand, are a set of reusable classes with no predefined behavior, interaction or flow of control (BOYD, 1993). They can be instantiated by a client and manipulated through it's Application Programming Interface (API) to achieve the desired effects.

While Frameworks and Libraries are different concepts, they both exist in a continuum based on the amount of control, structure and embedded knowledge that they have, with libraries being capable of displaying framework-like behavior and vice-versa (BOYD, 1993). As such, all concepts discussed from this point on about framework design and development are applicable to both libraries and frameworks.

PARSONS *et al.* (PARSONS *et al.*, 2006) highlight that frameworks are composed of two main parts:

- Frozen Spots: reusable pieces of software already coded in.
- Hot Spots: flexible elements customizable by users to meet concrete application needs.

When developing an application using frameworks, developers usually create hooks on hot spots through the implementation of hot spot subsystems, which encapsulate access to a hot spot by defining an abstract type with an interface for common responsibilities and different alternatives for the presented variable aspect (SCHMID, 1997).

The use of patterns and architectural abstractions can help discern if a set of classes are more closely tied to the application or the framework. As defined by JACOBSEN *et al.* (JACOBSEN *et al.*, 1997), application parts have specific functionality intended to be used as is, while framework parts represent a general parametrized aspect that can be changed or refined.

This flexibility and customization allowed by a framework can be characterized either through: a black box framework, in which the developer only knows about hot spots and a general description of it's use cases; a white box, where the developer has full knowledge of

the internal architecture and can build upon it (BOYD, 1993; PARSONS *et al.*, 2006). Table 1 shows a comparison between the two approaches:

	Black Box	White Box
Knowledge	No Knowledge	Developers have knowledge of the internal architecture and can build upon it
Hot Spots	Based on Inheritance	Based on Composition
Programming Languages	Limited to a single language	Can use more than one language
Learning Curve	Higher	Lower
Risk of Error	Higher	Lower
Flexibility	Lower	Higher
Control	Lower	Higher

Table 1 – Black-box vs White-box Frameworks

Overall, frameworks are usually developed with a gray-box approach, in which some hot spots use white box approach while others use a black box, with very few examples of actual pure black box or white box designs.

Frameworks that are Architecture-driven rely on inheritance for implementation, deriving new classes and overriding member functions (BOYD, 1993). Data-driven frameworks, on the other hand rely on composition and using different combinations of elements, with the combinations being made by the client and the framework deciding how they are combined (BOYD, 1993). Building frameworks an architecture-driven base with a data-driven layer makes them easier to use and flexible at the same time, with most frameworks offering both approaches to the client (BOYD, 1993). One such example is through component-based development frameworks, which contains well-defined units of reuse that can change their behavior after deployment without any changes to their implementation through the use of metadata (PARSONS *et al.*, 2006).

When developers are using frameworks to create their own applications, they interact with the API in order to build features. Having more intuitive code makes development more effective as it becomes easier to use, reducing the need to look at documentation resources, reducing the learning curve required to understand the code and it's relationships, allowing for more productive developers and higher quality software (PICCIONI *et al.*, 2013).

One approach we can use to create better APIs is through the use of Cognitive Dimensions of Notations, which work as a vocabulary of notations: a set of marks made in some medium, which may be visual or detected through other means (BLACKWELL *et al.*, 2001).

The concept allows the discussion of possible design problems that can arise in the creation, development, maintenance and evolution process.

As the frameworks change over time, they get more convoluted and present new problems for it's maintenance and evolution process. As such, it is important to follow guidelines to deal with this complexity and improve the structure of the framework (GURP; BOSCH, 2001): keep components small, with each having a single responsibility that's easier to reason about and being dependent on interfaces rather than concrete types; create interfaces that are focused on a single role, combined with the use of role inheritance to combine and compose different roles together to create an object; prefer loose coupling through events over delegation, and delegation of methods from other objects over inheritance; apply common specifications and solutions to problems already solved to avoid reinventing the wheel and easing the learning curve for developers; use standardized APIs for both standard and custom implementations, allowing for better dependency management over parts of the code; create documentation that's easy to comprehend, creating resources (e.g. descriptions, diagrams, code examples, sample programs) showing how the framework works and how to use it.

Overall, the goal with frameworks is to provide as much built-in functionality as possible around it's user requirements, creating solutions that are extensible and portable between projects while reusing the same base. By following good practices in API usability, standard design and coding guidelines, we can create solutions that maximize benefits and productivity for developers while minimizing potential client errors, steep learning curves and maintenance effort. As a framework matures over time, it's possible to derivate new frameworks from the original code, which encapsulate a specific subset of knowledge that is applicable in scenarios beyond the scope of the original framework.

3 STATE OF THE ART AND RELATED WORK

This chapter presents both the state of the art in VR games for PVI defined in the systematic mapping done in JÚNIOR *et al.* (JÚNIOR *et al.*, 2024), as well as the works directly related to this research, which is included as part of Inclusion Criteria (IC) discussion. For a complete breakdown of everything about the systematic mapping, we recommend checking the full work for reference, as we will focus on points closely related to this research.

3.1 Systematic Mapping

This exploratory research aims to understand how LRD concepts are trained in VR games for PVI. The research follows the methodological guidelines of KITCHENHAM *et al.* (KITCHENHAM *et al.*, 2009), WOHLIN (WOHLIN, 2014), and FAÇANHA *et al.* (FAÇANHA *et al.*, 2020).

The primary objective of this systematic mapping is to identify existing VR solutions that address LRD concepts, either explicitly or implicitly. This process aims to find reference works that can inform the development and evaluation of such applications. For this research, the selection of primary studies was guided by a search string, which was derived from a quasi-gold standard (ZHANG *et al.*, 2011). It is composed of four articles previously identified and selected by the research group:

- SIMÕES; CAVACO (SIMÕES; CAVACO, 2014) describes an educational mobile game designed to enhance orientation skills through virtual experiences, which positively impacts motor coordination in children with visual impairments.
- ZHAO *et al.* (ZHAO *et al.*, 2018b) presents an IVR experience that integrates a HMD with a cane used as a controller, facilitating the training of cane skills through audio and haptic feedback in a virtual environment.
- GONÇALVES *et al.* (GONÇALVES *et al.*, 2023a) explores the challenges and strategies employed by PVI to navigate and interact with gameplay elements and complex virtual environments in traditional digital games with inaccessible elements.
- WEDOFF *et al.* (WEDOFF *et al.*, 2019) introduces a hybrid environment game for PVI, emphasizing accessible physical activity and providing insights to guide the development of future accessible VR experiences.

3.1.1 Main Research Question

The primary research question guiding this study is: How do games and VR impact the development of LRD concepts for PVI?

3.1.2 Secondary Research Questions

The study addresses the following secondary research questions:

- **MRQ1:** For which types of low vision are the solutions designed?
- **MRQ2:** What are the target platforms and common techniques used in developing solutions for PVI?
- **MRQ3:** Which methodologies are commonly employed in the development of these solutions?
- **MRQ4:** How are the artifacts evaluated?
- **MRQ5:** What are the observed effects of VR on the development of LRD in PVI?
- **MRQ6:** What challenges are encountered in developing applications for this target audience?

3.1.3 Search String

The search string for this systematic mapping (see 2) was developed based on a quasi-gold standard, incorporating terms related to the four main research topics. The selection included both English and Portuguese terms to align with the language criteria of this mapping. The final string was refined through multiple iterations, testing various terms and expressions in search databases, and considering the research context, population, intervention, and results (WOHLIN *et al.*, 2012):

- **Games** (“Game", “Jogo", “Gamification", “Gamificação", “Game Design”, “Design de Jogos”): These terms were chosen to capture the researchers’ interest in games and gamification as they relate to LRD activities.
- **Visual Impairment** ("Deficiência Visual", "PDV", "PVI", "BLV", "Blind", "VI", "Visual Impairment", "Visually Impaired", "Low Vision"): These terms target the population of interest, including acronyms and variations that refer to visual impairments.
- **Virtual Reality** ("Realidade Virtual", "Head-Mounted Display", "RV", "Realidade Virtual", "VR", "Virtual Reality", “Ambiente Virtual”, “Virtual Environment”): These terms focus

on the research intervention, with "Head-Mounted Display" included due to its strong association with VR.

– **Left-Right** ("Direita-Esquerda" OR "Competência Espacial" OR "Esquerda-Direita" OR "Left-Right" OR "Percepção Espacial" OR "Right-Left" OR "Spatial Competence" OR "Spatial Perception" OR “Dominant Hand” OR “Mão Dominante”): These terms address the main concept and its synonyms. Notably, terms like "lateralidade" and "laterality" and "discrimination" were excluded based on early tests with the search string showing that they yielded more irrelevant results than useful research articles.

("Game" OR "Jogo" OR "Gamification" OR
"Gamificação" OR “Game Design” OR “Design de Jogos”)
AND
("Deficiência Visual" OR "PDV" OR "PVI" OR “BLV” OR
"Blind" OR "VI" OR "Visual Impairment" OR
“Visually Impaired” OR “Low Vision”)
AND
("Realidade Virtual" OR "Head-Mounted Display" OR
"RV" OR "Realidade Virtual" OR "VR" OR
"Virtual Reality" OR “Ambiente Virtual” OR “Virtual Environment”)
AND
("Direita-Esquerda" OR "Competência Espacial" OR
"Esquerda-Direita" OR "Left-Right" OR
"Percepção Espacial" OR "Right-Left" OR
"Spatial Competence" OR "Spatial Perception" OR
“Dominant Hand” OR “Mão Dominante”)

Table 2 – Search String used in the Systematic Mapping.

3.1.4 Search Databases

The search string was applied in the following search databases: ACM, IEEE, Wiley, PubMed, and Taylor & Francis. The databases were selected due to their proximity with this research’s topics, compatibility of it’s search feature with the search string and reliability.

3.1.5 Inclusion and Exclusion Criteria

For the selection process, we will use the following Inclusion Criteria:

– IC01: The work presents a VR game for PVI.
– IC02: The work provides relevant information on the ludic use of VR for PVI.
– IC03: The work includes information on how VR can be used to develop concepts related

to LRD.

- IC04: The work discusses challenges and techniques in developing VR solutions for PVI.

The IC were established based on the following goals:

- IC01 targets articles with VR games specifically designed for PVI to identify unique features and avoid redundancy with existing solutions.
- IC02 seeks articles offering ludic VR solutions for PVI that are not games, to gather important design references and considerations.
- IC03 focuses on articles that address LRD concepts using VR, including those that do not explicitly involve PVI but mention concepts beneficial for the target audience.
- IC04 explores challenges and techniques in VR development that can be adapted for this research, including VR techniques designed with PVI in mind that may be applicable in an LRD context.

As for the Exclusion Criteria (EC), they're designed to filter out papers that do not provide relevant solutions or techniques for the research topic:

- EC01: The work lacks a title, author, and/or abstract.
- EC02: The work is not classified as a short paper, full paper, or extended abstract.
- EC03: The work was not published in a conference or journal.
- EC04: The work was published more than 10 years ago.
- EC05: The work is not written in English or Portuguese.
- EC06: The work presents a solution that is beyond the feasible scope of this research.
- EC07: The work does not explicitly or implicitly cover LRD or spatial orientation for PVI.
- EC08: The work does not provide ludic digital solutions or games relevant to the research concepts.
- EC09: A preliminary review indicates that the work is outside the scope of this research.
- EC10: An in-depth reading reveals that the work is outside the scope of this research.
- EC11: An in-depth reading shows that the work does not meet any of the inclusion criteria.

This helps ensure that subsequent stages of the review focus on research topics that contribute meaningfully to the study.

For an article to be selected, it must meet at least one IC and not be disqualified by any EC by the end of the selection process. The selection process and qualitative evaluation of studies identified by the search string occurred in the following stages:

- **Identification:** This initial stage queried basic information such as the title, author, and

abstract. Any work that meets exclusion criteria EC01-08 was removed at this stage.

- **Selection:** During this stage, a skim reading of the "Introduction," "Related Works," and "Discussion" sections was conducted. To maximize coverage, "Related Works" and "Discussion" sections may be replaced by equivalent sections or treated as optional if the skim reading identifies a connection between the work and the research. Any work that meets exclusion criteria EC01-09 will be removed at this stage.
- **Eligibility:** This final stage involves an in-depth reading and analysis of the work to make a final assessment. For an article to be selected, it must not meet any of the exclusion criteria EC01-11 and must meet at least one of the inclusion criteria IC01-04.

To facilitate the procedure, we utilized Parsifal for labeling the works throughout the selection process and for analyzing general information during the "Identification" stage.

After the selection process, the chosen works were used to collect and analyze general information about the research, with the data categorized into four main areas:

- General Data: Includes publication details, authors, institutions, number of references, number of citations, and keywords.
- Research Methodology: Information on the methodologies employed in the studies.
- Experiment: Details on participants (including those with visual impairments and various vision profiles), and procedures followed.
- Results and Limitations: Findings of the research and any identified limitations.

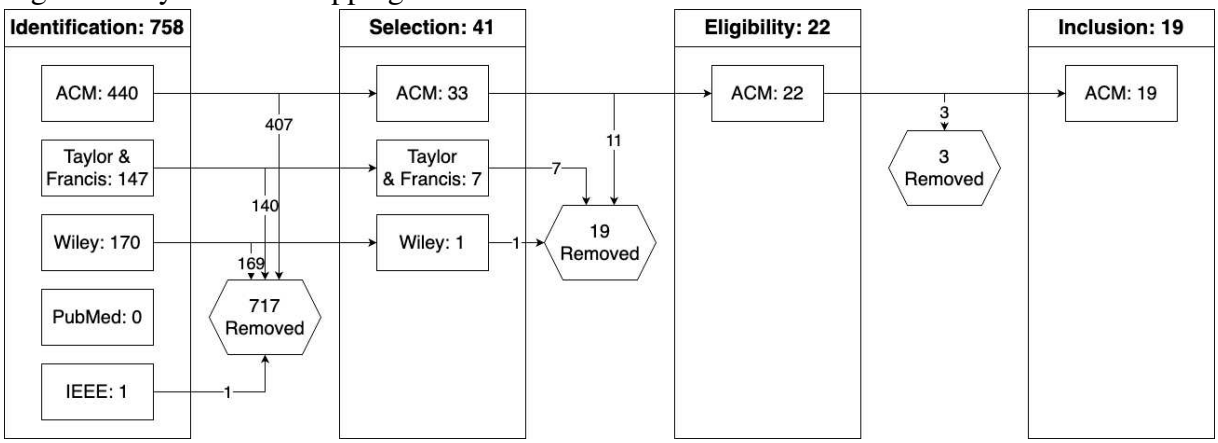
3.2 Results

Figure 3 illustrates the results of this systematic mapping. Ultimately, 19 studies were selected, all of which were sourced from ACM's Search Database. The spreadsheets containing the selected articles are available in a separate spreadsheet (Appendix A).

In the final list, both (ZHAO *et al.*, 2018b) and (ZHAO *et al.*, 2018a) refer to the same research presented at CHI 2018. After a detailed analysis, we determined that the content of each article was distinct enough to retain them as separate entries in the tables and for calculating percentages. However, for the purposes of discussion, they will be treated as a single research artifact (Canetroller).

Some key publications frequently covering the theme include CHI (5-26%), ACM Transactions on Accessible Computing (21%), and ACM SIGACCESS Conference on Computers and Accessibility (ASSETS) (11%). All these publications relate to the Human-Computer

Figure 3 – Systematic Mapping Results



Source: Author

Interventions (HCI) field, with the latter two specifically focusing on computing and accessibility.

The digital solutions identified were developed for the following devices:

- Personal Computers (Windows, Mac): 47%
- Smartphones (iOS, Android): 32%
- HMD and VR devices (Android, HTC Vive, Quest): 32%

It is important to note that while some studies addressed LRD concepts, the solutions were not explicitly or solely focused on LRD training, and the digital solutions presented are predominantly games and applications designed for PVI. However, Kreimeier and Götzelmann (KREIMEIER; GÖTZELMANN, 2018) is an exception, as their study did not have PVI as experiment participants. Other types of solutions identified include standards for using gestures to retrieve information in a virtual environment (PENUELA *et al.*, 2022) and the patterns used by PVI to explore virtual environments (GONÇALVES *et al.*, 2023a).

Although we found relevant results for this research, examples are limited: out of the 19 selected works, only 5 are identified as games, with only (WEDOFF *et al.*, 2019), (SIMÕES; CAVACO, 2014), and (NOROWI *et al.*, 2021) explicitly focusing on LRD. Nevertheless, other studies offer valuable insights for developing VR games for PVI with a focus on LRD concepts, even if they were not initially designed with these topics in mind.

Below we'll discuss each inclusion criteria, focusing on studies that are directly related to this research: for a complete breakdown, we recommend checking the paper of JÚNIOR *et al.* (JÚNIOR *et al.*, 2024).

3.2.1 Developed Works

The digital solutions presented are mainly composed of games and applications for PVI, with the exception of (KREIMEIER; GÖTZELMANN, 2018) where there was not PVI in the experiment. Other types of solutions found include standards for using gestures to obtain information in a virtual environment (PENUELA *et al.*, 2022) and the patterns used by PVI to explore virtual environments (GONÇALVES *et al.*, 2023a). Overall, 63% of the selected works use Unity to build their experiences, showing acceptance from the developers for using it in VR development.

While were able to find results related to this research, the examples are scarce: out of 19 selected studies, only 5 were identified as games, with only (WEDOFF *et al.*, 2019), (SIMÕES; CAVACO, 2014) and (NOROWI *et al.*, 2021) explicitly working with LRD. However, all of them bring useful contributions for developing VR games for PVI working with LRD concepts, even though some weren't initially designed with these topics in mind.

3.2.2 Games, Virtual Reality and PVI - IC 01

This section will present the solutions identified, highlighting important elements for building games for PVI. Initially, it's possible to notice that the games had a positive reception by PVI and had a positive impact in the skills trained.

Figure 4 – Development screenshot of Virtual Showdown

Source: (WEDOFF *et al.*, 2019)

Virtual Showdown (WEDOFF *et al.*, 2019) presents a digital adaptation of the sport Showdown, which is similar to air hockey. In this game, players wearing blindfolds use rackets to hit a ball and score goals against their opponents. Figure 4 illustrates an example of the game.

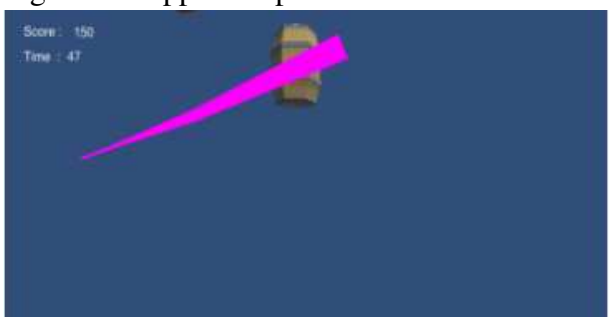
The game was developed as a hybrid environment solution, integrating Kinect v2, Nintendo Switch Joy-Con controllers, cardboard, and a table to define the physical space. Unity

was used to create the virtual play space, simulating a Showdown match while adjusting the physics to enhance gameplay and maintain fidelity to the real life version of the sport.

Kinect tracks the player's position on the table, while the bat's rotation is based on the player's body and wrist, which are fixed at specific angles to ensure stability. To assist players in inferring depth, the game employs panning and spatial audio cues to indicate the ball's location. The game features multi-modal feedback through the Joy-Con controllers: this feedback not only notifies players when they hit the ball but also guides their hands back to the table if they drift away, using different vibration intensities to convey the type of feedback required.

Following the hybrid environment approach, KIM *et al.* (KIM *et al.*, 2016) introduce Sonic Badminton, a prototype designed to make badminton accessible for both people with and without visual impairments. The prototype integrates sensors, electronic components, and a simulated environment to achieve this goal. Control of the game is managed by a Wireless Sensor Module attached to each badminton racket. The module captures the movements of the racket and transmits the data via Wi-Fi to a host server. The server processes this data to determine the racket's position (left, center, or right) and the corresponding swing motion (e.g., swinging from left to right). Apple Swipe (NOROWI *et al.*, 2021) is a mobile game inspired by Fruit Ninja: in it, players aim to slice fruits and avoid obstacles to achieve the highest possible score. Objects appear on the screen from various directions, each associated with a distinct sound that indicates its location, which helps players train Left-Right Discrimination. Figure 5 shows an example of how this application works.

Figure 5 – Apple Swipe



Source: (NOROWI *et al.*, 2021)

SIMÕES; CAVACO (SIMÕES; CAVACO, 2014) developed a mobile game featuring three scored challenges where players take photos of alien insects and complete tasks inside their spaceship using LRD abilities. The game employs spatialized audio and vibrations to convey

localization concepts such as left/right and front/back. Players use the device to locate interactive elements within the game, with the touch screen and gyroscope being used to determine the user's orientation and to interact with game elements.

Game	Main Features	OM Skills
Apple Swipe (NOROWI *et al.*, 2021)	Binaural Audio, Touch Controls, Progressive Difficulty	LRD, Aligned Commands, Clue Recognition
Sonic-Badminton (KIM *et al.*, 2016)	Binaural Audio, Motion Controls, Server-based Approach, Custom Physical Accessories	LRD, Body Image, Aligned Commands, Crossed Commands, Spatial References, Depth Perception, Clue Recognition, Motor Coordination, Dominant Side
Untitled spaceship game (SIMÕES; CAVACO, 2014)	Spatial Audio, Touch Controls, Gyroscope, Adaptive Difficulty	LRD, Body Image, Aligned Commands, Depth Perception, Clue Recognition, Motor Coordination
Virtual Showdown (WEDOFF *et al.*, 2019)	Spatial Audio, In-game Analytics, Progressive Difficulty, Haptic Feedback, Multi-modal Feedback, Hint System, Verbal Feedback, Custom Physical Accessories	LRD, Body Image, Aligned Commands, Spatial References, Depth Perception, Clue Recognition, Motor Coordination, Dominant Side

Table 3 – List of games working with Left-Right Discrimination concepts

Table 3 presents a summary of all games, including their main features and the LRD concepts identified.

Apple Swipe (NOROWI *et al.*, 2021) primarily focuses on LRD, aligned commands and clue recognition. These concepts are useful to achieve the game's objective of slicing fruit as it appears on the screen, with players swiping in the same direction as the spawn location.

In contrast, Virtual Showdown (WEDOFF *et al.*, 2019) and Sonic Badminton (KIM *et al.*, 2016) provide substantial opportunities to explore LRD concepts. These games incorporate:

- LRD with respect to the projectile's origin, trajectory, and potential destinations.
- Aligned and crossed commands in how players receive and return the ball or shuttlecock.
- Spatial references to understand the projectile's movement relative to the player's position and the desired direction for scoring.
- Body image and motor coordination through physical interactions, focusing on hand movements.
- Clue recognition through hint systems and multi-modal feedback.

– Dominant side considerations based on which hand the player uses to hold the bat or racket: for instance, WEDOFF *et al.* (WEDOFF *et al.*, 2019) found that participants achieved better scores when the ball came from the direction of their dominant hand or from a neutral position, such as the center of the table.

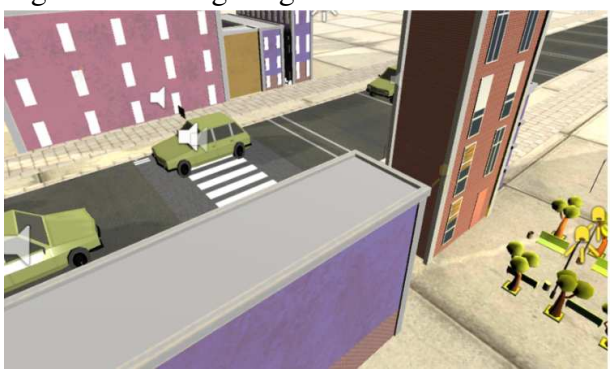
Simões and Cavaco's untitled spaceship game (SIMÕES; CAVACO, 2014) employs concepts across all three challenges: it trains LRD, aligned commands, body image, clue recognition, and motor coordination. In the first two challenges, players must locate and capture a creature, determine the direction and distance of sound sources, and navigate through the virtual environment while avoiding obstacles. The third challenge particularly emphasizes depth perception skills, requiring players to move around, search for objects, and complete tasks.

Although not presenting a digital game, GONÇALVES *et al.* (GONÇALVES *et al.*, 2023a) outlines important strategies used by PVI to play digital games that rely heavily on visual elements. Whether the game is intended for entertainment or educational purposes, it is crucial to implement mechanisms that allow PVI to actively engage with their skills. This ensures not only a balanced skill level between player groups but also an engaging and comprehensive experience.

3.2.3 Other Applications - IC 02

Although not games themselves, the solutions described in this section can be leveraged to develop VR games for PVI.

Figure 6 – Configuring a virtual world in X-Road



Source: (THEVIN *et al.*, 2020)

X-Road and AR Cube Hunter (THEVIN *et al.*, 2020) are VR applications for mobile devices designed, respectively, to facilitate classic OM tasks during classes and to explain audio-spatial concepts. The development of this application emphasizes several key features for using VR devices with PVI, like employing similar feedback mechanisms to represent analogous

objects and enabling users to locate elements in space.

As a tool for acquiring information from a virtual environment, ANDRADE *et al.* (ANDRADE *et al.*, 2021) developed an app that implements echolocation, a skill actively used by 20-30% of PVI. While the artificial echolocation provided by the app is useful for spatial orientation, the research indicates that natural echoes can be more effective for successful navigation tasks, as they offer a stronger sense of presence despite an initial learning curve.

VStroll (INDIA *et al.*, 2021) is a mobile app designed for PVI to virtually explore real-world locations and build a mental map of the environment. Users place their phone in their pocket and walk in place, utilizing the device's sensors to navigate through Points of Interest on the map. The app provides full autonomy through voice commands, allowing users to select paths at intersections and receive information as they move.

The preference for autonomous experiences is further supported by GONÇALVES *et al.* (GONÇALVES *et al.*, 2023b), who explored a virtual social space where PVI can interact with virtual agents in various scenarios. The findings indicate that users prefer the ability to navigate autonomously over the efficiency of the navigation methods used. Additionally, they value access to spatial information and how to approach and interact with other users in the virtual world.

3.2.4 Left-Right Discrimination and PVI - IC 03

In training LRD concepts, Virtual Showdown (WEDOFF *et al.*, 2019) demonstrated that participants performed better when the ball they hit stayed on the same side as their dominant hand or when it returned to the center of the table, suggesting that hand dominance significantly impacts the experience.

Apple Swipe (NOROWI *et al.*, 2021) shows how its fruit-cutting mechanics can be used to train LRD skills: players identify the direction from which fruit appears on the screen and make corresponding gestures to slice it, thus reinforcing left-right discrimination skills.

Utilizing life experiences of PVI can effectively inform the design of virtual environments. For instance, ANDRADE *et al.* (ANDRADE *et al.*, 2021) highlight how echolocation in virtual environments enhances the ability to distinguish object positions and obstacles over time.

Regarding feedback, audio is frequently used in many solutions to practice LRD. However, results from SIMÕES; CAVACO (SIMÕES; CAVACO, 2014) indicate that simple panning is insufficient for providing PVI with a strong sense of spatial presence. More complex

audio features, such as spatial audio and distinct sound techniques (GUARESE *et al.*, 2022), are necessary to a more effective experience.

In addition to sound, haptic feedback is also valuable for helping PVI understand their surroundings and detect objects in their personal space (ZHAO *et al.*, 2018b; KREIMEIER; GÖTZELMANN, 2018). Nevertheless, WEDOFF *et al.* (WEDOFF *et al.*, 2019) notes that players of Showdown prefer verbal feedback over simultaneous haptic and verbal cues, indicating a clear preference for auditory guidance.

3.2.5 Challenges and Techniques - IC 04

Analyzing the identified solutions reveals that diverse feedback mechanisms are a common feature, including audio (spatial or binaural) and tactile (haptic, tactile, vibrotactile, or force feedback). Other features contributing to robust solutions include TTS, gesture recognition, motion sensors, customization options, tips systems, and in-game analytics, which enhance both user experience and research outcomes.

MAY *et al.* (MAY *et al.*, 2020) discusses strategies for optimizing soundscapes in XR environments to avoid overwhelming users in high-density sound settings and to aid in cognitive mapping for PVI.

In a VR application designed for People with Low Vision (PLV), ODA *et al.* (ODA *et al.*, 2020) demonstrates how visual functions can be enhanced by applying transformations to the virtual scene, with results from a shooter game applying this technique showing improved performance for users with PLV.

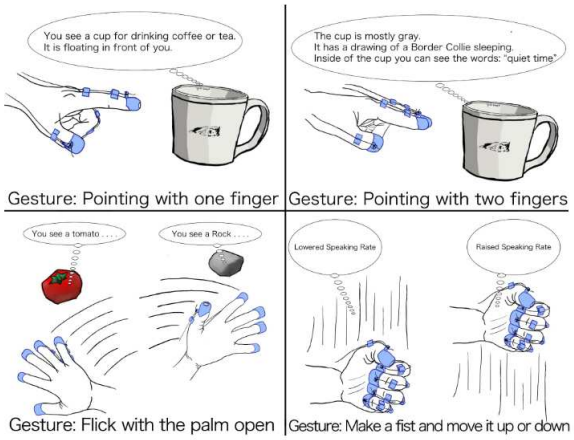
Through exploration and scene reconstruction experiments, ANDRADE *et al.* (ANDRADE *et al.*, 2021) illustrate how sound cues can help PVI acquire information about their environment, such as room size, wall openings, surface materials, and obstacles.

Using hand gestures, PENUELA *et al.* (PENUELA *et al.*, 2022) propose a standardized set of commands to allow PVI to obtain descriptions about objects or their surroundings.

X-Road (THEVIN *et al.*, 2020) emphasizes that achieving a minimal level of audio-visual fidelity is often sufficient for making virtual simulations accessible to PVI.

Accessible sound maps, developed by AZIZ *et al.* (AZIZ *et al.*, 2022), allow PVI to explore routes passively. The study outlines key design and usability criteria for such tools, including the integration of Auditive Icons (AI) to provide richer information during route exploration.

Figure 7 – Gestures as a tool for obtaining information in environments.



Source: (PENUELA *et al.*, 2022)

3.3 Discussion

In this subsection, we focused on detailing a subset of the results from Mapping Research Questions 1, 2 and 6, as they are the most relevant topics to this research.

3.3.1 Vision Profiles - MRQ1

MRQ1 Summary: (i) Participants included individuals with and without visual impairments; (ii) PVI samples mostly made up of Totally Blind (TB) or PLV individuals; (iii) Some studies do not differentiate participant numbers based on the type of visual impairment; (iv) Including people without visual impairments in research focused on PVI requires a meaningful role that involves comparison or collaboration between the two groups.

Based on the data provided by the authors, the presented classifications define various types of visual impairment, enhancing our understanding of the research to create more tailored interventions for PVI.

At the same time, including people without visual impairments in research for PVI can threaten the research's validity, as they are not the target audience. Therefore, it is important to have a well-defined reason for including them in the research.

When working with games for PVI, knowing the type of visual impairment of your audience also affects which techniques should be used to relay information and improve interaction, locomotion and navigation in the game, as some approaches are more effective when working with certain groups (e.g. high-contrast modes for PLV), ordering tasks based on the most effective features.

3.3.2 Technologies Used - MRQ2

MRQ2 Summary: (i) Predominance of solutions for computers, smartphones and HMD; (ii) Use of TTS, spatial and binaural audio, customization features and multimodal feedback; (iii) Unity as the engine of choice for development

The digital artifacts developed (games, virtual environments, IVR simulations and other apps) demonstrate the use of commercially available devices to create VR solutions for PVI. By using established technologies for PVI such as TTS and spatial audio, we can improve the learning curve for users, leading to a better experience. Choosing Unity as a development engine underscores it's versatility and widespread adoption, enabling seamless deployment across multiple devices and operational systems.

3.3.3 Challenges and Limitations - MRQ6

MRQ6 Summary: (i) Audio panning is not enough to represent a three-dimensional space; (ii) Avoid overloading PVI mentally with the information presented; (iii) Ensure representativeness in the spectrum of visual impairment for the sample of participants; (iv) Importance of supplying information that is both representable and customizable; (v) Balancing a realist representation of the virtual world while keeping it distinct from the real world to avoid confusion; (vi) Reduce the friction of interaction between PVI and the virtual world;

To ensure a good experience, it's important to consider several criteria for developing VR for PVI, such as using audio solutions for three-dimensional spaces, building the information architecture to offer representative information that can be customized by the user without causing a mental overload, and balancing the fidelity levels of the experience. Another challenge is fulfilling the needs of various types of visual impairment, making use of their unique skills and adjusting the experience accordingly based on the system's interactions.

3.4 Threats to Validity

The procedure was discussed with the research group before execution, but due to time constraints, it was carried out by only one of the authors. Measures were implemented to address potential issues with this process. Articles from a quasi-gold standard were selected to guide the mapping process in order to meet the stated objectives. However, it is possible that

chosen works influenced the lack of explicit references to LRD. To mitigate this issue, the search string was expanded to include related concepts, combined with meticulous reading to identify any implicit mentions of LRD.

Another potential threat to validity concerns the search string itself. Works that address LRD skills implicitly might use different terminology, or articles might use different definitions or acronyms for PVI. To address these concerns, we identified relevant terms based on articles previously reviewed by the authors. For instance, terms like “Spatial Competence” and “Dominant Hand” were chosen based on LRD studies. Additionally, the search string was reviewed by another author before initiating the systematic mapping process. Nonetheless, reliance on terms from the quasi-gold standard may have led to omission of important terms, such as “handedness”, which we discovered during the mapping process.

The Inclusion Criteria and Exclusion Criteria developed might also pose a threat to systematic mapping, as relevant articles could have been excluded in initial stages due to insufficient information in the title, abstract, or skimmed sections. To minimize this risk, criteria were established to select relevant articles from the beginning of the process, with a focus on Introduction, Related Works, and Discussion sections. These criteria were intended to swiftly identify the research’s purpose, foundation, and contributions.

Another identified threat was the risk of including articles that do not fit the mapping criteria. To address this, the procedure included a retroactive application of Exclusion Criteria, with additional criteria for articles that do not meet the inclusion or exclusion criteria. The Inclusion Criteria were also designed to ensure that the selected articles have valuable and applicable contributions to the research.

3.5 Chapter Conclusion

This chapter explored how VR can be utilized by PVI, with a particular focus on developing LRD skills, as well as details about frameworks and how the Unity engine works. The VR studies reviewed showcase immersive experiences where PVIs interact physically with the virtual environment, leveraging their unique skills and providing a rich, engaging experience that is transferrable between the virtual world and the real one. While the examples are limited, they demonstrate approaches for progressively training skills, offering benefits such as increased engagement, motivation, physical activity, and social inclusion.

One problem we encountered during this process was that solutions listed by this

systematic mapping are unavailable both in binary and source code, reinforcing the opportunity and importance of making solutions available to help developers create games working with the concepts they present. With this in mind, we decided that our research should follow an approach similar to the work of MARIA *et al.* (MARIA *et al.*, 2023), but applying it to the field of OM and LRD. In other words, this works attempts to create an open-source solution that acts as a starting point for creating VR games for PVI working with OM and LRD concepts.

While Unity is a closed-source software, challenging the possibility of creating a fully open source software solution in this situation, all of the package's source code is licensed under the MIT license[1], allowing developers to adapt it to fit their needs and make it compatible with open-source game engines based on C#.

[1] https://opensource.org/license/mit

4 METHODOLOGY

For this work, we plan to adapt methodologies for framework development with a top-down approach based on the elements analyzed on Table 3 to create and evaluate EdKit, a new open-source library condensing these effects.

Based on the work of Bosch et al. (BOSCH, 1997) and Van et al. (GURP; BOSCH, 2001), our approach focus on the creation of a framework to enhance the process, offering predefined interactions and decoupling components, letting developers adapt it to best fit their use cases. The evolution of the library is expected to change elements on the API and resemble more of a framework later on: as such, the design process must be prepared to deal with changes as new requirements arise (GURP; BOSCH, 2001).

Figure 8 shows a Business Process Model and Notation (BPMN) diagram of how the methodology was modeled and will be applied for this research:

4.1 Domain Analysis

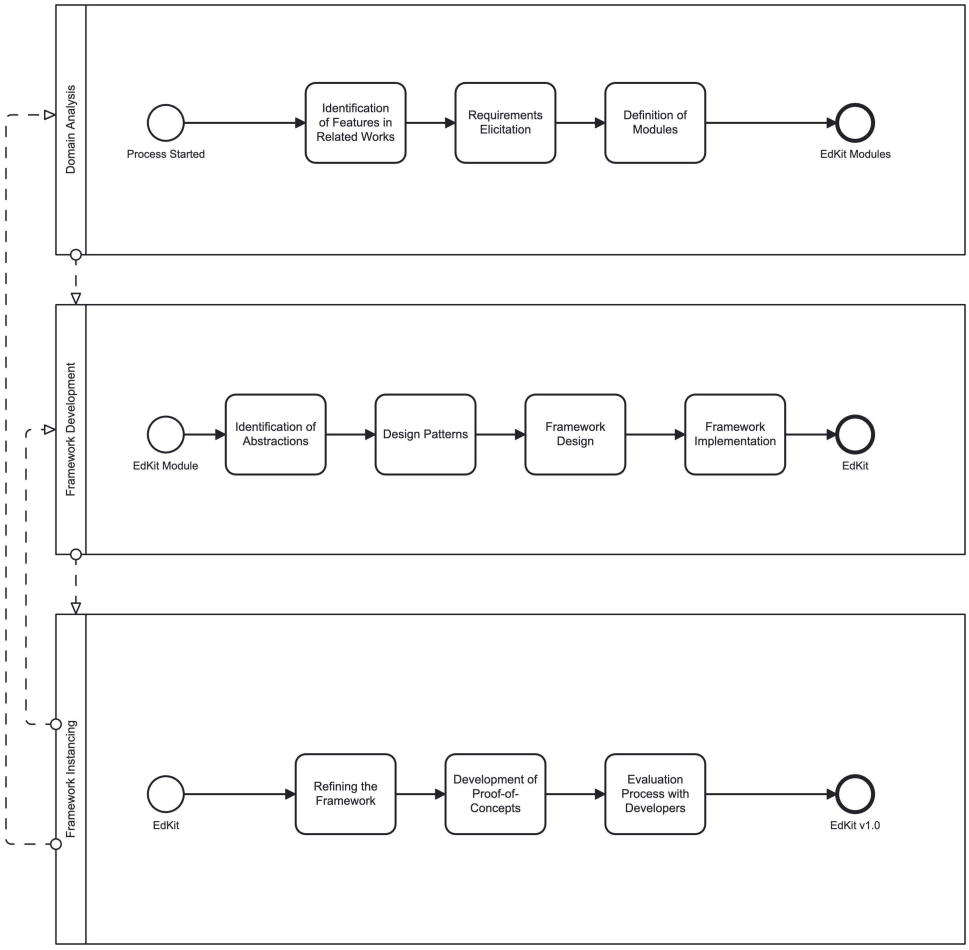
First, a **Domain Analysis** was made, starting with the *Identification of Features in Related Studies*, which was based on features listed in Table 3 to decide which areas should be tackled. As a domain framework, developing EdKit from scratch allows it's design to reduce the amount of work required to implement applications in the domain area (GURP; BOSCH, 2001).

Afterwards, we selected what will be implemented by package through an *Elicitation of Requirements*, in order to define which concepts will be represented. As EdKit is focused on the development of LRD concepts for PVI in an OM environment using VR, we focused on requirements that allow developers to have a workflow that can be applied to these contexts:

- Express the following LRD concepts extracted from 3: LRD, Body Image, Aligned Commands, Crossed Commands, Spatial References, Depth Perception and Clue Recognition.
- Offer information for PVI through more than one channel of information, such as verbal description, sound or even visual when factoring in low vision.
- Give developers a way to analyze the player's actions in the game in order to understand it's performance and create custom tools for OM instructors to guide future activities.
- Make a package that's compatible with VR game development and easy for game developers to integrate in their games.

We then make *Definitions of Modules* that will fulfill these requirements, which are

Figure 8 – BPMN Diagram of the Research Steps



Source: Author

placed within a single package with all modules bundled together. This approach was chosen initially for ease of development: as the API matures and new use cases arise, these modules can become standalone and be branched off into their own packages. It's important to note that using two or more frameworks together can be prone to composition problems (GURP; BOSCH, 2001), so the creation of EdKit should take into account that it is just one of the many packages a game developer will use to make their games.

4.2 Framework Development

After defining the modules to be developed, the **Framework Development** step allows us to define what elements should be created for each module.

After defining the abstractions present in the library, we can define which *Design Patterns* can be applied in order to describe architectural aspects of parts of the framework, improving it's generality, parametrization and specificness (JACOBSEN *et al.*, 1997). One

example of this in EdKit was the use of the Component pattern to design certain features in each module that can be configured in the Editor.

Then we go through the process of *Framework Design*, defining the structure of the API, the types that compose the package and it's relations. For this work, we opted to have the "UnityEngine" standard library as the single dependency, as we focused on a small solution using only the essential types and covering a large space of use cases in the knowledge domain (NYSTROM, 2014). We also avoiding depending on other packages the overhead cost, technical debt and learning curve that can come with other external codebases.

Following the design phase, the process of *Framework Implementation* happens in Unity and progress was versioned through Git to register the changes for the source code over time, as means of documenting the evolution process of the API and it's design changes.

4.3 Framework Instantiation

After developing an implementation for EdKit, it went through **Framework Instantiation** in order to evaluate it's effectiveness in fulfilling the requirements presented and gather it's acceptance from game developers. The process started through the iterative process of *Refining the Framework*, becoming more robust and focused on it's design.

After cleaning up the structure, EdKit was applied directly in the *Development of Proof-of-Concepts*, creating three small games using the package to show how it can be used to develop games for PVI with LRD. By creating these smaller games as described in Chapter 6, we looked into critical points of the API, handling edge cases and identifying new approaches for a better coding experience.

EdKit was then applied in a *Evaluation Process with Developers*, as described in Chapter 7, in order to determine the learning curve of integrating the package into their development workflow, focusing on developers with prior Unity experience.

4.4 Chapter Summary

This section presented the methodology used to develop and iterate over EdKit. By using the package in a game and showing how the package can be used by developers in a real project, new features, bugs, edge cases and feature requests will arise, which then are applied back into the development process.

5 EDKIT

This chapter presents EdKit's design, detailing decisions made and references for all it's modules. Section 5.1 presents the general approach to EdKit, with Sections 5.2, 5.3 and 5.4 detailing each of EdKit's modules.

5.1 General Information

Based on the requirements gathered in this process, we designed EdKit[1] as a custom Unity package that can be used by game developers to aid in developing VR games training Left-Right Discrimination concepts.

We decided to develop EdKit for Unity based on the solutions described in Chapter 3. As for the version, we chose Unity 2022.3 as it was the most recent long-term support version at the time of the decision. EdKit's file structure follows Unity's package layout conventions(TECHNOLOGIES, 2023) to simplify the process of updating to newer versions.

EdKit acts as a gray-box domain library, encapsulating a vertical slice of features containing expertise related to LRD and OM concepts, being mainly focused on: corporal image (worked through poses and gestures); the distinction between left-right, up-down and front-back; orientation; clue recognition (aided by a feedback system).

The package uses as a mix of scripting approaches:

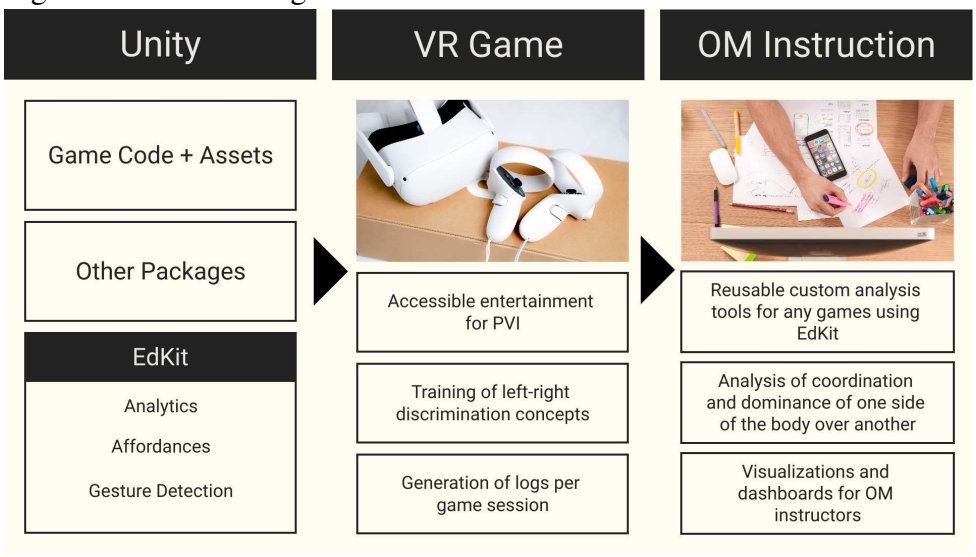
- Architecture-driven: inheriting from classes and implementing interfaces to create library extensions.
- Data-driven: using functional programming concepts and data structures from Unity.

The modules for EdKit were chosen through a domain analysis process of solutions listed in Table 3, with the aim of offering a complete package for LRD games for OM with the least amount of elements required. Section 8.1 details which concepts are embedded explicitly and implicitly in the package. Based on the requirements identified, Figure 9 shows a feature diagram of the package's architecture, which is defined into three main modules:

- **Gesture**: defines how objects are positioned in relation to references in space and how to identify a gesture from the player's position and use it.
- **Feedback**: defines representations for an affordance and how it can be applied to generate multimodal feedback.

[1] EdKit is available as an open-source package on Github under the MIT license: https://github.com/MartonioJunior/EdKit

Figure 9 – Feature Diagram for EdKit

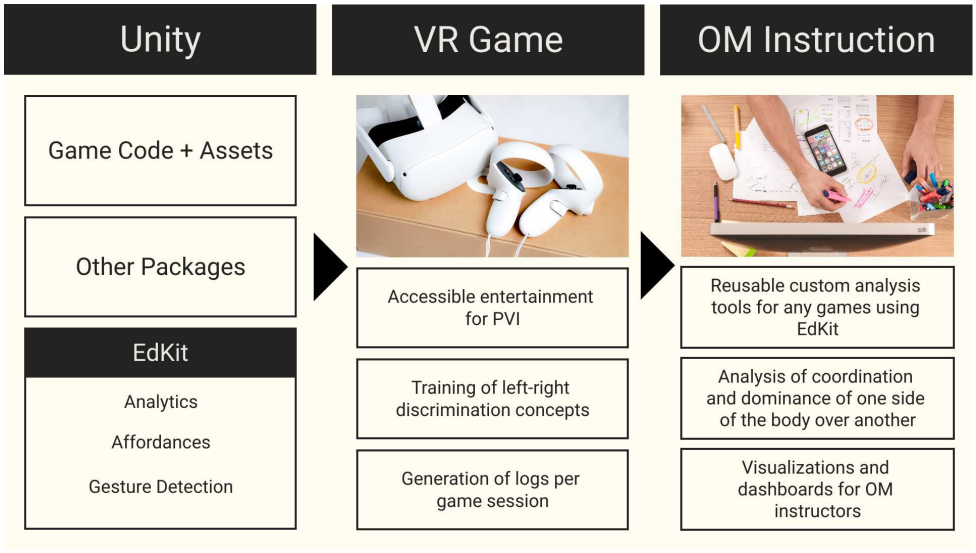


Source: Author

– **Analytics**: keeps track of events that happened in the game and stores them in logs available to OM instructors.

The package was designed with XR development in mind, but it's components can be used in non-XR games as the interpretation of LRD concepts does not depend directly on input devices.

Figure 10 – EdKit in the game development process

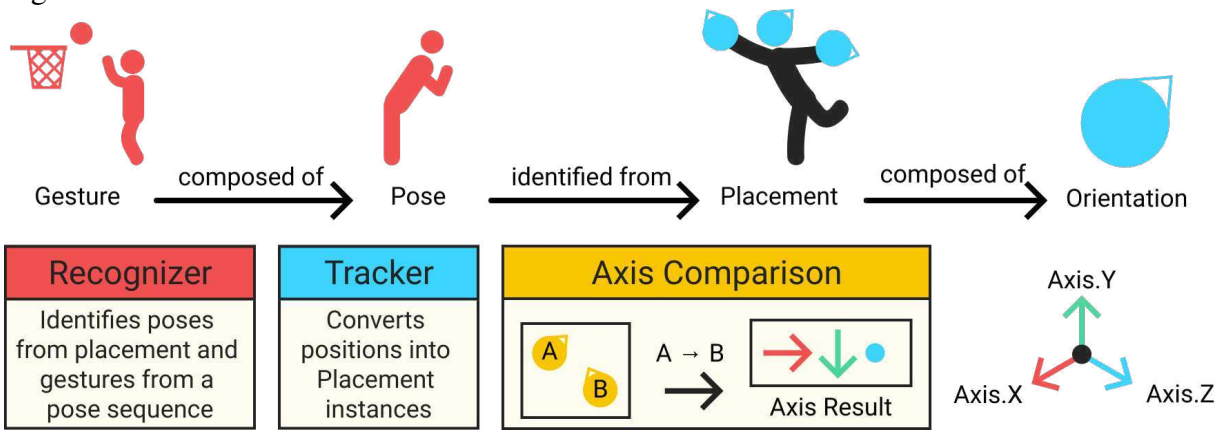


Source: Author

Figure 10 shows an initial proposal of a user flow for how EdKit would appear in the development process, having the following key stakeholders in mind:

– Game Developer/Programmer: Develops the games being made for PVI, has knowledge

Figure 11 – Structural Model for “Gesture”



Source: Author

of how the Unity engine works, creates tools to aid OM instructors and have direct contact with EdKit, as they use it together with other packages and solutions to create their games and custom tooling.

– OM instructor: Helps OM students in their rehabilitation process, uses games and other tools to improve their lessons: doesn’t have direct contact with EdKit, but uses logs created by EdKit in custom tools to see how their OM students are performing.
– OM student: PVI that is going through the rehabilitation process and likes to play games: their play sessions in games developed with EdKit are saved as logs that can be sent to their OM instructor(s).

Game developers can use the entire package or just parts of it in other related contexts, such as games for PVI that are not designed for VR in mind. They can also use it with other packages to implement their games in Unity, allowing programmers to focus on developing other features, such as gameplay or accessibility settings.

5.2 Gesture

The gesture model is based on the ideas behind spatial references and body image highlighted in Section 2.1. In order to identify gestures and poses made by the player, we decided to focus on basing our approach in an egocentric reference point, making it easier for PVI to perform gestures independently of where their body is pointing towards.

For the gesture model, we encapsulate spatial references in order to determine relationships in a scene. Figure 11 shows how you can compose together elements to create a new gesture.

Orientation is a structure containing the relative position and rotation for an object based on another in a determined moment in time. The type can be used as a reference point to place other Transform components and create new Orientation instances in relation to itself: with 3 of them, one can define a **Placement** data structure for any game actor (e.g. players, enemies, Non-Playable Character (NPC)) and identify poses.

When comparing a spatial reference against another, we use enumerators to give a signed comparison between two objects (e.g. "Axis.X.Left" indicates that a object is to the left of another). Inspired by Kim et al.(KIM *et al.*, 2016) and Wedoff et al.(WEDOFF *et al.*, 2019), each axis is separated into 3 regions as described in Table 4, with an extra case for a Null Object for inconclusive or invalid comparison cases (e.g. position of objects in a 2D space does not use the Z axis).

Axis	Negative Value	Neutral Value	Positive Value
Axis.X	Left	Center	Right
Axis.Y	Below	Neutral	Above
Axis.Z	Back	Body	Front

Table 4 – Definition of terms for Axis

Programmers can compare any two Orientations together in order to obtain an **Axis** data structure which contains the result of this comparison: when making the comparison, deadzone values can be passed in indicating the maximum delta in an axis for it to still be a neutral value. All neutral values were also given different names to describe the center position so developers can identify which Axis is associated with messages logged in a debug console.

Game programmers can use these types alongside basic position or rotation operations to store information related to vector comparisons in a coordinate space, and using them to create spatial relationships between elements of their games. Some examples of this applied to OM include: an alignment task, where a player must align two objects of the same type to face in a single direction; telling where an object is located in a scene based on another location; logic puzzles, where players need to organize objects in a sequence and clues can request users to organize in a certain way, following reference points (e.g. flag must be placed to the left of the book).

By combining Orientation instances together in a Placement and interpreting it, developers can determine a **Pose** out of a single Placement and a **Gesture** out of a sequence of them. Each Pose or Gesture identified in EdKit is stored as a **Event**, containing:

- Object: the pose or gesture associated with the event
- Data: information about the event
- Score: Value detailing the similarity between Object and Data
- Timestamp: Value indicating when the event happened in-game, using the internal clock for the scene.

Poses are a way to score how an actor's Placement relate to a specific arrangement, employing a "Bring Your Own Algorithm" strategy in which the programmer defines how poses are scored. They're responsible for defining a detection algorithm for a Placement instance, returning a score indicating the accuracy in relation to the Pose. This approach offers greater flexibility in implementation, allowing you to choose the evaluation method—whether it's a standard algorithm, a Machine Learning (ML) model, or a custom algorithm tailored to your game's specific needs.

As a guideline, scoring a pose yields a positive value when there is some resemblance to the actual pose across one or more Orientations, while a negative value or zero indicates that a pose do not fit the Placement at all. Poses identified in EdKit are registered as **Pose Events**.

There are two main ways to create a pose in EdKit:

- Implement a new type in code that conforms to the "IPose" interface.
- Instantiate a new "Pose" in code by passing the algorithm as input.

Another approach that was considered in development was to create a new pose directly within the Unity Editor using Scriptable Objects, with a default algorithm of bounding boxes for each *Orientation*, where developers could define reference bounding boxes for limb positioning and rotation, with scores ranging from negative values up to 1, being represented as follows:

- Negative Value: Position or rotation is outside the bounding box.
- Zero: Position or rotation is on the edge of the bounding box.
- Positive Value: Position or rotation is within a bounding box's defined ranges.

In the end, we decided to not follow through with this approach due to usability concerns and high friction when evaluating the system in the pilot test for the evaluation.

Furthermore, poses created in code can be easily overridden, as developers can insert additional code before and after choosing an algorithm. By allowing total flexibility and control over the process, this enhances functionality (e.g incorporating custom analytics) and enables complete replacement of poses when unit testing other components in your game.

By composing one or more poses together in a sequence, you get a **Gesture**: an interpretation given to a sequence of Pose Events, generating a score based on it's accuracy. Just like poses, you can either create them as a new type conforming to "IGesture" interface or by instancing a new "Gesture", following the same scoring guidelines.

Gestures can be created using similar methods as Poses and designed to enable creation of complex motion inputs by leveraging existing implementations of detection for poses and combining them together. We also considered a ScriptableObject approach to creating Gestures, in which you could put the Poses you've created in an order to create a gesture. While this approach had better usability, it was reliant on the non-scripting version of Poses and therefore, were both listed are "deprecated" components inside the package.

Recognized gestures are stored in **Gesture Events**, which include a sequence of Pose Events, identified Gesture, and timestamp of detection. To implement pose and gesture tracking for an actor using EdKit, it is necessary to add two components to your game:

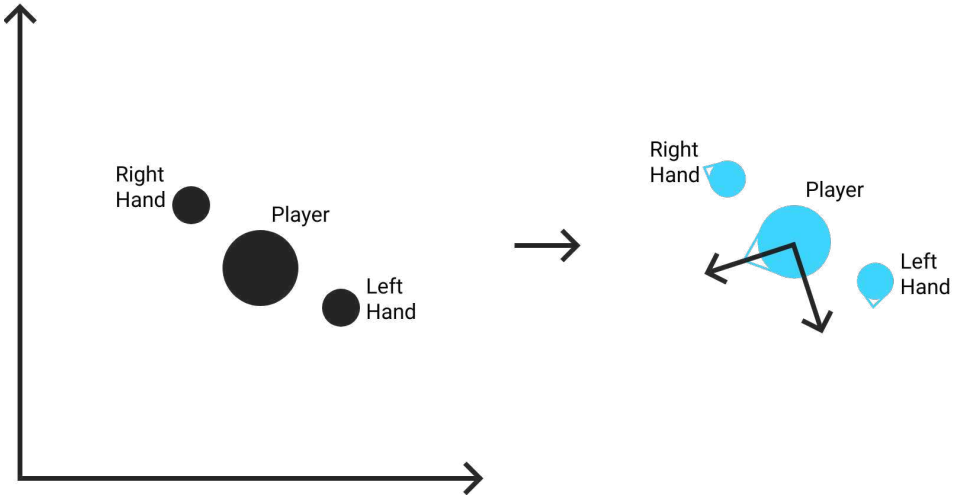
- **Tracker**, component which returns a *Placement* for an actor's Transforms
- **Recognizer**, component which determines poses and gestures from *Placement* instances.

Tracker is a EdKit component that converts "Transform" instances representing an actor's hands and head position in a scene into a *Placement* suitable for use in pose and gesture recognition. The process of capture happens in the "Sample" method, which triggers an event passing an obtained placement for the actor that can be used by other components. This method is automatically called at the "Update" phase of Unity's PlayerLoop, but developers can disable the component and use another approach to trigger it manually. The capture process relies on four Transform references related to the actor: Head (or equivalent); Left Hand (or equivalent); Right Hand (or equivalent); and Actor's Origin (e.g., XR Origin).

Figure 12 shows the conversion process involves changing coordinate systems for the actor's positions and rotations: this new system's origin is defined by the head's forward vector relative to XZ plane and origin's up vector, transforming positions from World to Local space and saving them as a Placement instance. By creating a egocentric reference point players can perform gestures even if they drift away during gameplay, eliminating the need to reposition themselves in physical space.

While position tracking can be prone to errors and calibration issues, VR devices like the Meta Quest 2 provide excellent precision, supporting six degrees of freedom. EdKit also mitigates these challenges by not relying directly on input devices; instead, it utilizes Transforms

Figure 12 – Converting Unity Transforms to Orientation



Source: Author

to represent the virtual locations of the player's limbs. By handling information stored in Transforms rather than directly processing motion device inputs, developers can create custom components that interact with these devices, which can be particularly useful for NPCs or for enhancing tracking quality with less precise motion devices. This flexibility allows developers to effectively manage inaccuracies, whether by offering more precise input alternatives or improving motion detection.

Recognizer is a EdKit component that is responsible for Pose and Gesture detection. As Pose Events are registered in this component, a Recognizer registers them in an ordered buffer to track the temporal sequence of identified Poses, enabling detection of both Poses and Gestures within the system. When each pose is registered in, Recognizer follows these evaluation steps:

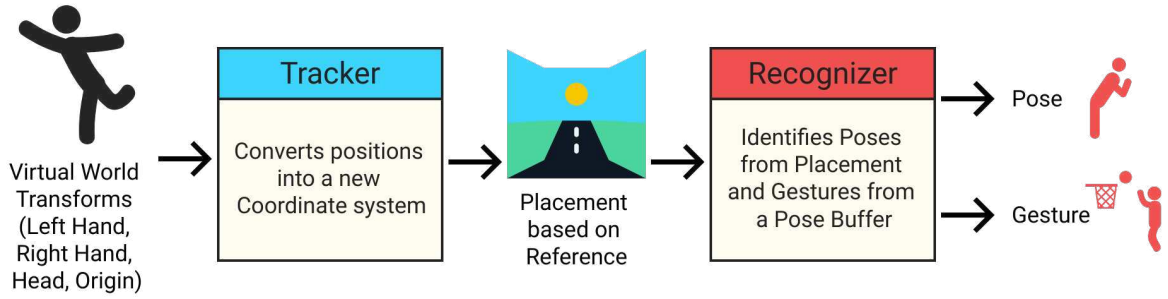
- *Pose Selection*: Using a pool of potential poses, Recognizer selects the highest-scoring pose for a newly added Placement.
- *Pose Event Creation*: A new pose event is created with the selected pose, notifying other components through UnityEvents and adding it in the buffer.
- *Gesture Selection*: Evaluates the buffer's current state against a pool of possible gestures, selecting the highest-scoring one.
- *Gesture Event Creation*: A new gesture event is generated, notifying other components of the identified gesture using UnityEvents, and the pose event buffer is reset.

Poses and Gestures are evaluated internally through **Event Evaluators**—custom data structures that manage a pool of possible candidates and determine minimum score thresholds for evaluation. The Recognizer employs two Event Evaluators: one for poses and one for gestures,

which can be tuned independently to refine control over which events are considered relevant.

When adding a Gesture created in Unity, its corresponding poses are also included in the pool of potential recognizable poses. Figure 13 shows how an actor in a VR space can be interpreted into poses and gestures.

Figure 13 – Gesture Recognition in EdKit



Source: Author

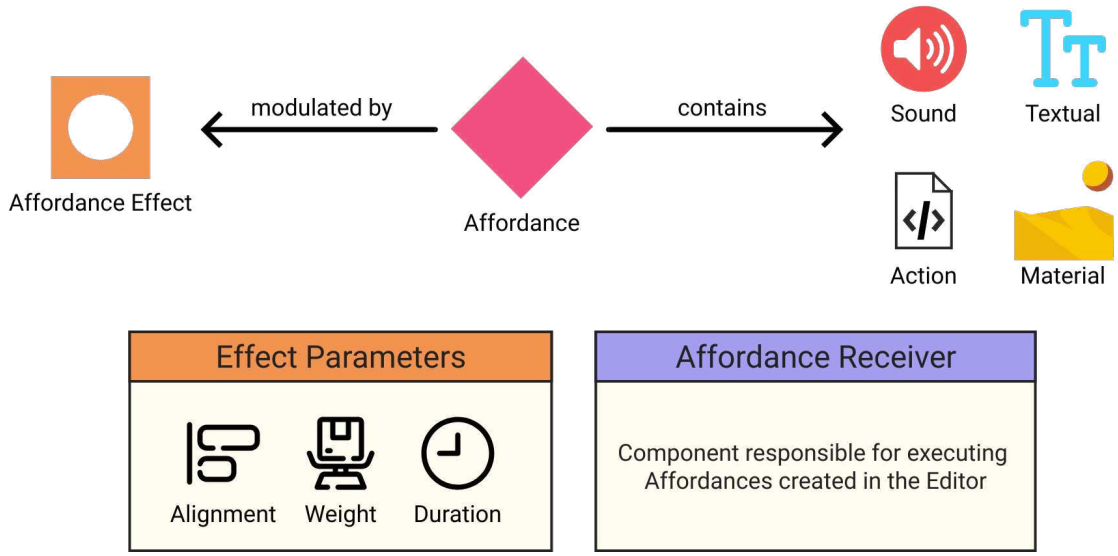
With poses and gestures, developers can create VR games that enable players to execute a wide variety of actions without pressing a single button. This approach not only provides flexibility in selecting and implementing features but also facilitates effective unit testing and this modular approach empowers creators to tailor their solutions to fit specific game requirements.

5.3 Feedback

When discussing the creation of VR games for PVI, RIBEIRO *et al.* (RIBEIRO *et al.*, 2024) emphasize that various locomotion mechanisms offer distinct advantages and player preferences. Importantly, these approaches can all be accessible for the player, provided that players receive the necessary cues to understand them. For the feedback system, we have adopted a structure inspired by NOROWI *et al.* (NOROWI *et al.*, 2021), WEDOFF *et al.* (WEDOFF *et al.*, 2019) and KIM *et al.* (KIM *et al.*, 2016), in which developers can encapsulate the same information with multiple representations.

Figure 14 shows the structure of the module, which is entirely based on **Affordances**: a data structure that depicts multiple representations of feedback for an information. As the information is stored together in a single object, it makes it easier for developers to deliver multimodal feedback through different channels, as well as provide redundancy and alternative stimuli to give to players. Typically, they encompass up to four types of data representations:

Figure 14 – Structural Model for “Feedback”



Source: Author

– *Sound*: This can include audio clips, mixers, effects, or other sound data representations.
– *Text*: This may involve strings, contextual text types (such as localized text or smart strings), or any other textual/verbal data representation.
– *Visual*: This can encompass colors, materials, shaders, animations, meshes, or other visual data representations.
– *Action*: Custom code that receives an Affordance Effect as input.

Like gestures and poses, affordances can be created in code by conforming to the “IAffordance” interface or by instantiating a new “Affordance” object: with this, you can freely choose which types will represent sound, textual and visual feedback, as well as add custom execution code based on your needs (e.g. haptics feedback).

For the latter approach, instead of passing the data object directly as a parameter, it accepts functions that return the data objects you want to execute at a given time. This approach allows developers to implement custom logic within an action, enabling dynamic changes to the data object being used (for example, altering the sound clip played based on whether a combo sequence is active or not).

In both approaches, the “Update” method can be used to run the affordance directly from code, with the programmer having the responsibility of updating it’s execution over time. This method allows for the implementation of custom actions and loop-driven feedback mechanisms that require updates over time, such as haptic feedback or dynamic audio panning.

When executing an Affordance, developers must also specify an **Affordance Effect**:

a data structure representing spatial information for a LRD context. Affordance Effects provide contextual information about how an Affordance is applied in a given context, being composed of three main parameters:

- *Alignment*: A normalized value that describes a spatial relationship based on a reference axis.
- *Weight*: A normalized value indicating the effect's impact along each axis, with values ranging from 0 (no impact) to 1 (maximum impact).
- *Duration*: A floating-point number that specifies how long the Affordance should last, measured in seconds. A duration of -1 means the effect will play indefinitely until replaced, while a duration of 0 triggers the update method only once.

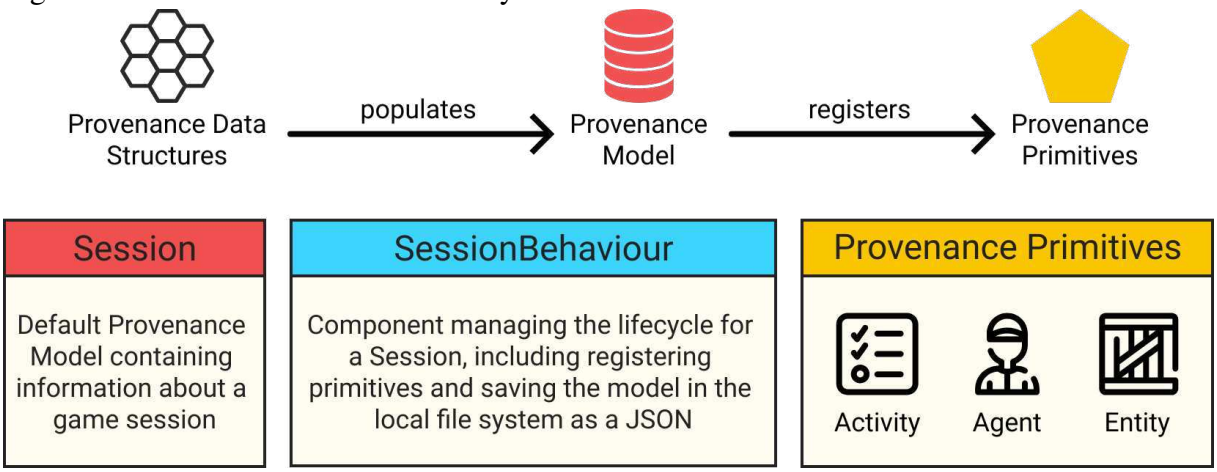
For instance, settings like "alignment = -1, weight = 0.2, duration = 1" can indicate that the ball is coming from the left, is far away, and the feedback will play for one second.

Another approach available in EdKit is through ScriptableObjects, in which developers can define an "AudioClip" reference for sound feedback, a "Material" reference for visual feedback and a string for textual feedback. Running these affordances can be done either using the same methods as the ones created in code or via an **Affordance Receiver**: a component that manages the lifecycle of ScriptableObject-based affordance, ensuring that they're only executed one at a time. If a new instance is applied, the currently executing Affordance is overridden. When you apply an affordance to an affordance receiver, the component fires events for each representation when you start, update or finish executing. The affordance effect used can be passed in when you apply the affordance or use the default effect for the component instead.

In summary, "Feedback" enables developers to define and execute stimuli based on the game's context, with Affordances providing a valuable source of informational redundancy that enhances PVI through multiple sensory channels. Examples of what can be done include playing a verbal description and sound cue when an object is grabbed, changing an object's texture and playing a jingle for low-vision interactions, or adding haptics and sound effects to simulate heavy machinery, with sensations adapting based on proximity to the object.

This design approach was chosen to consolidate multiple representations into a single data structure (for example, a bell sound and the description "ball to the left" to indicate an incoming ball) and let it be responsive to the context of the game. This allows the same Affordance to be reused in different areas of the game, conveying the same stimuli while enabling variations based on context.

Figure 15 – Structural Model for “Analytics”



Source: Author

5.4 Analytics

Another important pillar of EdKit is how it can be used by OM instructors to gauge how their students are performing while playing a game and aid in their teaching process. As we aim to also give developers tools for tracking game events performed by PVI players and save them for future use, our analytics model is inspired by the PROV-O ontology ((W3C), 2011; MARQUES *et al.*, 2024).

This design allows us to log various OM metrics, such as the position of the player’s hands, time spent executing activities, hand movement (left versus right), and the accuracy of interactions with objects approaching from either side. Since each game has its own unique data types and structures for logging attributes, our solution is designed to be both well-structured—organizing logs into sessions using the PROV-O ontology—and flexible, allowing attributes to utilize any serializable type.

Figure 15 shows the structure of the module: taking base in PROV-O ontology, EdKit contains three primitive structures for analytics:

- **Entity**: Elements with a persistent identity (e.g. documents, files, data types).
- **Agent**: Actors, groups or systems responsible for performing tasks.
- **Activity**: Represents operations and actions performed on entities. Can be associated to an Agent and/or Entity.

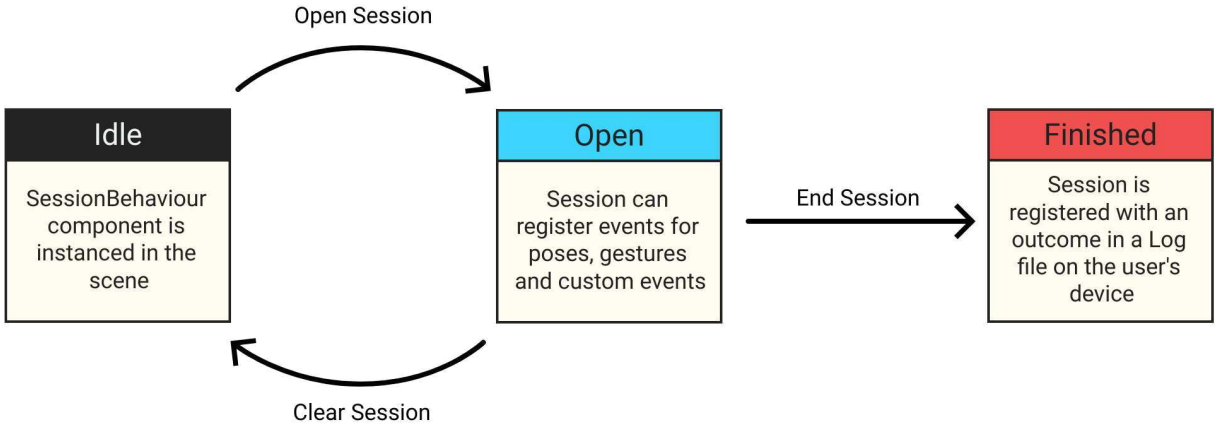
In EdKit, all analytics primitives can be stored in a **Provenance Model**, which registers information about a given context (e.g. player session). We leverage serialization in provenance models to store entities, agents, activities and their attributes. All the information

registered in a model that is serializable can be saved in a JSON format, allowing developers to integrate specific information about their game state and save them in a log file for future use.

Developers can also create their own **Provenance Data Structures** by conforming to the “IProvenanceData” interface, being able to control how it is registered in the provenance model: to do so, the programmer decomposes their data model into one of the primitive structures of analytics, then registers then directly into the model.

While EdKit lets developers create their own model using the “IProvenanceModel” interface, developers have the alternative of creating a **Session**, which is a provenance model that encompasses all in-game information for a played session, including date and time of play, user, scene, outcome for a session, and details about participating agents, entities, and activities.

Figure 16 – Analytics Session Lifecycle



Source: Author

The lifecycle of a Session works as shown in Figure 16:

- Opening a Session: Developer initiates a session by passing identifiers for user and scene.
- Event Registration: While a session is active, events are registered using any available methods.
- Closing the Session: Developer concludes the session by providing an outcome, which is then saved into a log file.

Analytics sessions are managed by the “SessionBehaviour” component, which handles the entire lifecycle and acts as a proxy for the Session structure, enabling the registration of new structures through the same methods available in the “IProvenanceModel” interface.

With this system in place, developers can leverage analytics to create LRD games for PVI, allowing an OM instructor to monitor player performance and progress. Developers can

issue patches to balance gameplay based on player actions and identify which elements players engage with in the game. By maintaining a log of player activities, valuable information can also be extracted and analyzed using custom tools, such as dashboard visualizations (MORAES *et al.*, 2023), in order to inform their decisions.

5.5 Chapter Conclusion

This section showed EdKit, a custom Unity package for developing VR games for PVI. The package provides a quickstart approach for gesture detection based on the player's reference points, along with a structure for multi-modal feedback and an analytics model for games with in-game events stored in a log that can be read by custom tools.

Chapters 6 and 7 detail more about the work done for EdKit through it's instances, improvements to the user experience and the impact of the package for game developers

6 DESIGN AND DEVELOPMENT

In this chapter, we present basic examples showing how EdKit was be used to create the framework instances: Section 6.1 details the common structures for the project; Section 6.2 presents “King”, a simple reaction game where the player must make the commands informed by the game, with the goal of obtaining the highest score possible; Section 6.3 details “Boat”, a puzzle arcade game in which the player must perform gestures to launch orbs into space and chain them together to earn points; and Section 6.4 presents “Ed”, an endless runner where the player goes through on-rails environments to get the best score while balancing time, energy collected and energy used.

These games were developed for the Meta Quest 2 line of devices with the goals of being accessible for PVI to use, training LRD concepts through motion gestures in VR and using all of EdKit’s modules.

6.1 Implementation Details

All instances are part of a shared Unity project that included the EdKit package and other general utility packages for physics, sound and more to aid in the development of the solution, simulating the use case presented on Figure 10. The extra packages included do not overlap with EdKit’s features in any capacity, and serve only as a way to show how packages can interact and be extended by developers when used together with others in a project, as development is focused on the instances themselves and improving the study artifact.

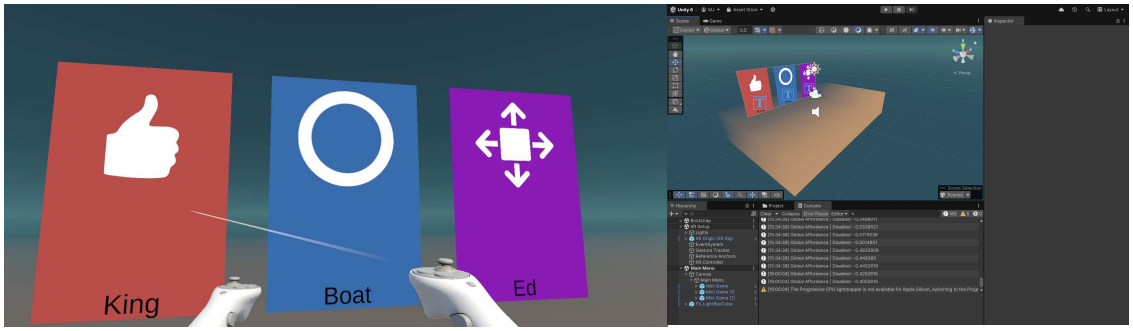
The scene structure for the project is based on a multiple scene workflow, with values being passed between scenes through the use of Scriptable Objects. It is divided as follows:

- When the game is booted up, it initializes a “Bootstrap” scene to decide which scenes must be loaded in and what components must exist through the lifecycle of the game.
- All configuration for VR input is located in the “XR Setup” scene, which also contains an EdKit gesture tracker component to get the player’s placement.
- To select the games, a “Main Menu” scene is created with the interface to select a game.
- Each game contains it’s own set of three scenes, containing: game logic, visuals and user interface.

The games were packaged together, with all entries being accessible through a main menu revealing the list of games available. Figure 17a shows an example of this main menu.

Figure 17 – In-Development Screenshots of Menu Screens

(a) Main Menu (b) Project (Main Menu)



(c) Start Screen (d) Pause Screen (e) End Screen

Source: Author

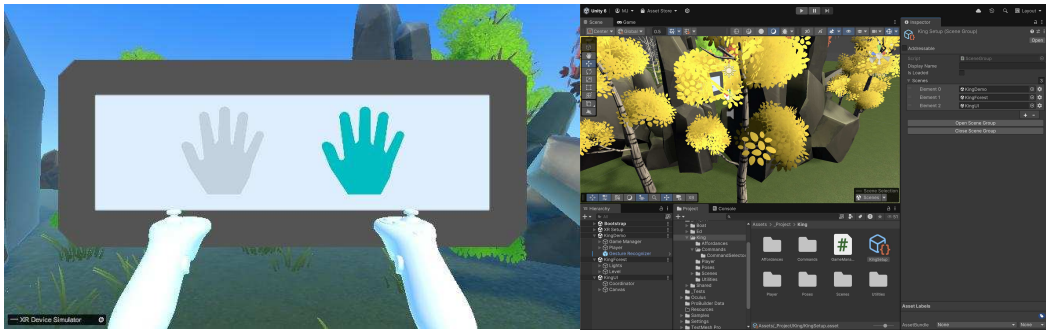
As we're making the games with People with Visual Impairments in mind, hovering over each game makes a sound jingle and a TTS message starts playing, explaining what the game is about before selecting it to play. The player can select a game by pressing the trigger button over a game, while Speech can be interrupted at any time by pressing the grip button on the left controller. All games initialize with a starting menu screen prompting the user to begin by pressing a button, with the background representing the stage where gameplay will take place. Figure 17c shows an example of this starting screen in one of the games.

By pressing the same button again, the gameplay is resumed with an affordance being played indicating that the game has been resumed. When a game ends, the player is shown a results screen containing a message informing the score for this session: results are messaged through an affordance object with a description that is updated after the game finishes. Figure 17e shows an example of a results screen.

6.2 King Game

King is a single-player VR game in which you must perform commands correctly in order to earn points and stay in the game. It was developed as a mini game made with the goal of evaluating the pose recognition system and it's related APIs while also showing a game that uses all three modules together. The rules are inspired on a common activity performed by PVI with OM instructors, in which they need to both understand requests given out by the TTS voice and

Figure 18 – In-development screenshots of “King”.
(a) Game (b) Project



Source: Author

perform them.

Figure 18 shows a screenshot of this small game, in which the goal is to make the highest score possible by doing commands to earn points. With the game, PVI can test the following skills:

– Plane Slicing: distinguishing between left and right members.
– Body Image: understanding the position of their hands and head to raise flags correctly.
– Clue Recognition: listening to verbal commands and sound jingles to perform their actions
– Motor Coordination: raising their upper members correctly.

We focused on making the game with the lowest amount of requirements possible possible using all EdKit modules.

6.2.1 How It Works

When the game begins, the player receives a command to be performed within the time limit: each command has it’s own Affordance, containing a request to be spoken via TTS and a visual representation representing the request that will be used by the user interface. Then, the player must perform one of the required command by raising either the left or right hand and keeping the other lowered. Commands can be changed until time runs out, letting players take their time in making the right one, as well as chance to change options if they feel they did a wrong gesture.

When time runs out, the player gains a point for a correct command and a new request is generated to be followed: an affordance is played with a jingle, the message “Correct!” being spoken via TTS and the user interface becoming green representing that the player got it right.

If instead, the player makes a wrong command, they lose a life and must make another attempt to get the command right: an affordance is played with another jingle, the message “Incorrect!” via TTS and the user interface becomes red representing that the player got things wrong. When the player opts out of making any pose, it will not remove a life from the player, but also not award any points.

This risk/reward system was decided upon in order to give the experience an arcade experience, in which the player increases their score as much as possible while avoiding losing as many lives as possible. The game keeps going until all lives are lost or the player decides to quit, ending with the score obtained by the player.

6.2.2 Using EdKit

When beginning a new game, each possible “Command” wraps a pose from EdKit’s Gesture module and can be registered directly into a gesture evaluator using an extension method. The game has a command selector component that is responsible for selecting the next one to execute (while avoiding repeating commands) and evaluating whether the pose event received is the current one. After receiving a pose event by wiring the event from the gesture tracker directly into the registration of the gesture recognizer, it evaluates and fires the respective event for a correct or wrong command performed, which are wired as follows: when the player executes the right command, the player earns a point and the selector chooses a new Command to be executed; when the player fails, they lose one life and can try again to perform the right Command.

Meanwhile, EdKit’s Affordance module was used to represent information that’s passed directly to the player, such as: which command the player must execute. They were created without scripting, using the ScriptableObject-base Affordance to define a textual description, a sound clip, a material and custom haptics to play the introduction telling how the game works and how to execute the commands, ending message revealing the final score, as well as tell the player what command is next and when it executed the right or wrong command. This game instance uses Unity events together with EdKit’s components to execute the required Affordances in the affordance receiver.

The game contains a Game Manager component that controls the lifecycle of the game that works together with EdKit’s Analytics to integrate the Session in the SessionBehaviour component with the game loop, create Activity instances associated with the Player Agent when it gains score or loses a life, as well as work as an Entity object that stores the score obtained,

number of lives and time elapsed.

6.2.3 What We Learned

By developing “King”, we were able to prove that EdKit’s features work as intended, while also noticing pain points in how poses are structured in the architecture. The following list contains issues encountered and changes made to EdKit’s API in the development of ‘King‘:

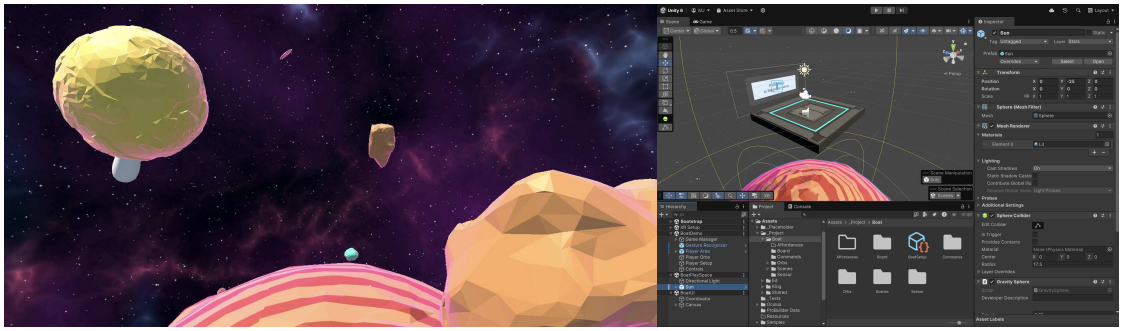
- Added default values in methods to reduce verbosity when using it in a default use case (e.g. “Orientation.ComparePosition” method compares against the origin of the new coordinate system when no value is supplied)
- Improved use of gesture evaluators with the gesture recognizer and in the Unity Editor.
- Unified the implementation of gesture and pose events to use a common Event structure: before, the package had separate types for each event type.
- Added a conversion of Axis enumerators into affordance effects to create the latter by taking advantage of the spatial relations, including one that returns a signed normalized value out of an Axis enumerator.
- Added the capability of an affordance receiver to stop the Affordance from being executed.
- Fixed issues related to the serialization of attributes when saving a Session into a Log.
- Added new utilities for an activity in Analytics to allow association with analytics entities.
- Removed the capability of a provenance data structure to register itself into a models, as it may cause endless recursion as a cyclical dependency.
- Added events to “SessionBehaviour”, as well as a debug toggle for saving Logs in the File System in order to avoid clutter when using Play Mode in the Editor

An alternative to integrate gestures in this game using EdKit include creating it’s own gesture recognizer component for commands, that either connects to the tracker directly or foregoes it completely. Commands can also be created as it’s own standalone types, making for a more customizable approach for the recognition algorithm.

Another way to implement feedback using EdKit for this sample game would be by creating a custom affordance type through the “IAffordance” interface, being able to define which representations to use and even add additional information that can be sent to another component.

While analytics in EdKit does not offer many alternative implementations outside of custom provenance data structures or models, it is possible to create your own session

Figure 19 – In-development screenshots of “Boat”.
(a) Game (b) Project



Source: Author

lifecycle by creating a custom component to manage a “Session” instance without the use of “SessionBehaviour” (e.g. when the session is suspended and will be continued later).

6.3 Boat Game

Boat is a single-player VR arcade puzzle game for PVI in which the player throws stars in order to rack up points and get the highest score. Figure 19 details a screenshot of the game, showing the player area on the sun’s orbit with stars flying around him.

Boat was developed as a traditional game in order to test how the Feedback structures can be used to let PVI play games with greater spatial complexity. The rules are inspired in casual puzzle games such as “Tetris” and “Suika Game”, adapting the experience to use more non-visual elements to inform the player about the state of the play area and actions it can perform.

With the game, PVI can test the following skills:

- Plane Slicing: distinguishing between left and right stars to launch appropriately, determining the location and trajectories of stars in orbit.
- Aligned and Crossed Commands: Launching a star forward can both follow the same orientation or the user may opt to cross it up.
- Depth Perception: The player must notice how stars are moving around them.
- Clue Recognition: listening to sound jingles and verbal information in order to know where the objects are and make a decision on where to throw them.
- Motor Coordination: making a throw movement gesture to launch stars into orbit determines the direction they will take.
- Spatial References: understanding where stars are located and their trajectories in relation

to the player.

The goal with this game is to make an arcade puzzler that is accessible for PVI to play, focusing on a simple but engaging experience that does not heavily depend on the visuals.

6.3.1 How It Works

After starting the game, the player appears in an empty field standing at a fixed spot in the sun's orbit with an introduction message told via TTS about what is the goal of the game.

Then, stars start spawning over time in the play area, which is composed of a left side and a right side and both can only hold one orb at a time. Every time a star is spawned, an affordance is played with a sound jingle telling the type of orb spawned and the panning on the sound telling where it did spawn (left or right).

This approach was chosen so that the player has a well defined and limited set of tools, giving more importance into when and where the orb is thrown as it will take a bit of time before getting another. Another goal with this is so players are always working their plane distinction skills while playing.

The player can throw stars with a gesture, each with it's own type and an initial amount of energy. Stars ride around the sun's orbit until they hit another object in their trajectory. When a star collides with another, one of the following things can happen:

- When colliding another star of the same type, they will be combined by adding together their energy amounts. The player earns points every time a combination happens and an affordance is played with a sound jingle indicating it was absorbed.
- When colliding another star of a different type, they will bounce against each other, changing their trajectories with no changes to their energy amounts. An affordance is played with a jingle indicating the collision happened.
- When colliding with the sun, the star is repelled out into orbit again. An affordance is played with a jingle indicating the orb was repelled back into orbit.

These are micro goals that player can do, giving an incentive for the player to throw orbs and attempt to earn points.

While both stars and sun have a gravitational pull to attract other flying objects to themselves, the latter only has it when it is smaller, being meant for a homing effect when they're more unlikely to interact with other stars. As stars gather energy, they increase in size and become easier to collide with other stars. Once a star reaches the required thresholds, it can end

up creating either a Supernova or a Black Hole depending on the levels for each outcome and based on it's respective type

If a star has enough energy to become a Supernova, it must collide with another star that fulfills the same requirements to create a new Supernova. When a Supernova happens, it clears all stars and Black Holes in the area around where the collision happened, earning the players the sum of the amount of energy in each star removed as points and with an extra amount per Black Hole destroyed in the process.

If a star combines with another star and crosses the required threshold for a Black Hole, the star transforms itself into one, staying in a fixed spot on the sun's orbit. Just like the sun, it has it's own gravitational pull, but it also has the ability to absorb other stars, increasing in size after each one.

The game ends once the player is consumed by the Black Hole, with the result being based on the amount of points obtained by the game.

6.3.2 Using EdKit

For this game, we implemented two new poses for throwing something to the left or to the right of the player and reused two existing poses from "King", related to raising their respective left and right hands. These poses were composed into two throw gestures, each representing an orb that could be launched by the player. This decision was made to show how a common pose can be implemented only once and reused for multiple contexts.

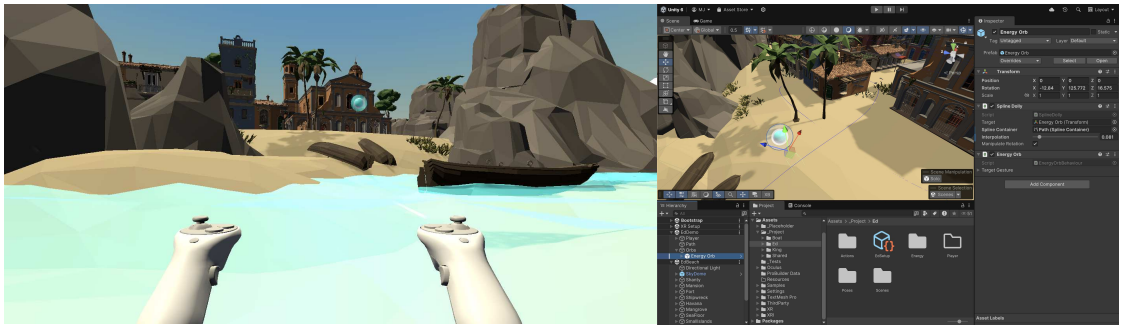
In terms of affordances, there's one affordance for each type of orb floating around, which is executed when the orb enters the vicinity of the player area, playing a sound jingle for the flying object and the type of orb being spoken via TTS for redundancy. Each affordance is positioned spatially using an effect based on where it is entering from, aiding in the panning and volume for the sounds.

For analytics, we had activities for when the player sets a score, with the orbs and the final result being entities and the player as the agent.

6.3.3 What We Learned

By developing "Boat", we were able to understand better how EdKit's affordances can be improved to better fit the needs of a game, with one of the main changes to EdKit's API being a rework of how affordances created in the Editor are played using an affordance receiver:

Figure 20 – In-development screenshot of “Ed”.
(a) Game (b) Project



Source: Author

– A default effect can be defined in the component, which is useful for when binding the method to Unity events in the Editor.
– The component can now help with affordances that require constant updates, passing in to “AffordanceData.UpdateAffordance” a modified effect containing only the delta time for the current update cycle. When the time for an affordance runs out, they are removed from the update queue and stop their execution.
– While the component still uses a “fire-and-forget” approach, it can now manage affordances to avoid repeated executions of the same affordance too quickly, using the duration from the effect as a cooldown. Affordances can also now be removed from the receiver both to stop them and to quickly clear them for another execution
– When starting, stopping or updating an affordance, the component now fires a event per feedback type, passing in the data representation as a parameter

Beyond these changes, we also added a custom inspector to the Gesture Recognizer component in order to make it easier for developers to debug the code associated with poses and gestures.

6.4 Ed Game

Ed is a single-player VR on-rails runner where the player must perform gestures to cross through a trail to reach the goal at the end. Figure 20 shows a screenshot of the game, in which the player is running through a field.

Ed was designed as a game based on motion controls, in which the player has a set of more interesting choices it can make with more physical input and activity. The initial design for EdKit as a package originated from this game idea, as concepts heavily emphasized the use

of gestures together with feedback to inform PVI of the game's events and their movement in the stage. The rules are inspired by other fitness games such as "Ring Fit Adventure", using gestures to let the player interact with the scenario and handle how fast it goes.

With the game, PVI can test the following skills:

- Plane Slicing: being able to distinguish the location of an orb as it comes in, being either to the left or right, above or below in order to collect it.
- Body Image: understanding the location where the hands should be and how the arms are organized to avoid clashing them while moving them around.
- Aligned Commands: performing gestures where the hand moves in the same plane as it is located.
- Crossed Commands: performing gestures where the hand moves to a plane different from the one it is located.
- Depth Perception: paying attention to orbs as they get closer to the player.
- Clue Recognition: listening in to sound cues to know when an orb is nearby and understand PoIs in the environment.
- Motor Coordination: moving the members accordingly between multiple poses in a short span of time.
- Dominant Side: choosing to use one hand over the other when grabbing an orb, as well as giving preference to collecting orbs spawning in a specific region on the trail.
- Spatial References: paying attention to PoI in the environment to know where you're located in the trail.

The goal with this instance is showing how flexible and dynamic the gesture system can become, as the implementation used for poses can be composed together to create a gesture, as well as how information from gestures performed can be used in games.

6.4.1 How It Works

After starting the game, the player appears on a trail moving at a fixed default speed and must perform actions by combining poses together to create a gesture associated with an action. While adventuring through a trail, movement speed is based on the player's actions, which can speed up or slow down progress. After executing any action, the game executes an Affordance representing the action that was executed, containing a jingle and a material for the player. If the player is performing no action that influences their movement, the player's speed

slowly changes towards the default amount, accelerating or decelerating as necessary.

The list of actions available for the player include:

- Accelerate: the player raises and lowers their hands above their heads, alternating between left and right to gain speed on the trail. The amount of speed gained depends on the time spent to perform the gesture, as well as if the alternation followed a slow approach (raise left $\rightarrow$ idle $\rightarrow$ raise right $\rightarrow$ idle) or a fast one (raise left $\rightarrow$ raise right). With this, the player could improve their completion time while also risking having less reaction time for collecting orbs, having to choose when it should speed up.
- Decelerate: the player opens their arms wide, moving the left hand to the left and right hand to the right. As long as this pose is performed, the player loses speed over time and is able to move even slower than the default speed, having more time to figure out the location of orbs and collect them. With this, the player can keep moving at a more limited speed and lose momentum in exchange for increasing the reaction window for collecting an orb.
- Stop: the player crosses their arms in order to completely halt their movement. As long as this pose is performed, the player stays fixed in place, being able to take in the environment and the elements around them. With this, the player can exchange all of their momentum for taking their time in understanding the location of the orbs, as well as being able to prepare to collect them.
- Super Speed: the player puts their arms forward and moves them to their back in order to gain the maximum amount of speed possible. As long as the player's hands are kept behind them, the player maintains momentum at the cost of losing energy collected previously at a fixed pace. The stance ends when the energy runs out or the player leaves the pose, reducing their movement back within the movement speed range. With this, the player can spend energy collected to get even better completion times and having the smallest reaction time window for collecting orbs out of any action.

While going along on a trail, the player can find orbs next to it that can be collected by touching them. They appear at fixed locations on the trail, as players passing through must move their hand in one of the following directions: top left, top, top right, right, bottom right, bottom left or left.

When collected, they give the player some energy that can be used, and there's no limit on how much energy it can be held. An affordance is played with a jingle that represents

the energy being absorved, with the panning being adjusted based on where the orb was located in relation to the player.

The game ends when the player reaches the end of a trail or when the time limit expires. After finishing the trail, the player's performance is evaluated based on three main factors: completion time, energy collected and energy used. Actions were balanced to be based on what goal the player is trying to achieve:

- A player going towards a better completion time will aim to collect and use as much energy as possible, going extremely fast and requiring quick reaction times to collect more and not run out during this scenario. As it does so, this type of player will focus on performing acceleration gestures as much as possible, and may lose out on some orbs along the way, spending all the energy they get on Super Speed whenever possible.
- A player aiming for the highest amount of energy collected will opt to have more time to collect orbs, going more slow and even decelerating and stopping in certain spots to analyze the layout. As it gains energy, it will use Super Speed to have more time to spare collecting orbs while also being able to complete the trail within the time limit.
- A player dedicated to not using energy will focus on performing gestures in order to complete the stage within the time limit. It will also will not be as dependent on orbs, collecting them as they come and likely being the player type with the most amount of energy at the end of a trail.

This approach not only accounts for player types aiming for different goals, but also looks at players with different skill levels, as beginners can start playing aiming for one goal type and change it based on their needs and developments.

6.4.2 Using EdKit

For this instance, we implemented seven poses for the player to perform:

- Run Left: raises the left hand above the head without crossing it and keeps the right hand down.
- Run Right: raises the right hand above the head without crossing it and keeps the left hand down.
- Yipee: raises both hands above the head while not crossing them and keeping a certain distance from each other.
- Still: hands are below the head. Used as a default fallback pose in the game.

- Airplane: the player extends their arms in a T-shape.
- Block: the player crosses their hands (left hand to the right, right hand to the left).
- Launch: the player puts both of the hands backwards.

These poses are used to compose the game's five gestures, with Run Left, Run Right and Still are used to create the two gestures related to the Accelerate action. The other gesture are a single pose, each relating to their specific action: Airplane for Slow Down; Block for Stop; and Launch for Super Speed.

When the game begins, the player hears an introduction affordance, containing a TTS talking about the goal of the game. For each action, there's a respective affordance to indicate that was the command executed, composed of a jingle, a material to indicate a shift in speed and a TTS message. When you collect an orb successfully, an affordance also plays indicating that you successfully collected the orb, with a jingle, a material and a TTS message.

For the analytics of the game, we have the following: Energy, Orb and Outcome are entities; action executed, orb collected and set energy as activities; and Player as an agents. When collecting an orb, the activity is associated to the orb collected while player is associated to actions executed (in the case of Super Speed, the Energy entity is also attached to the action).

6.4.3 What We Learned

While this game did have the initial implementation ideas that came to became EdKit, we decided to have it last as an instance to compare the progress made by other instances towards the final design of the package. Therefore, there aren't as many changes in the API as other instances, but we still found some important improvements by completing "Ed":

- Added "Any" as a case to allow orb zones to work with orb zones.
- Added methods to allow "Orientation" instances to: generate an accuracy score based on a bounding box; quickly create a new orientation by placing another one in relation to itself; new comparison methods for position in rotation that use offset and scale values to create a bounding box for comparison.
- Deprecated "Affordance", as the "IAffordance" interface already fulfilled it's purpose better and allowed for clearer implementation of data representations.
- Improved debug for poses and gestures in the Editor by adding better string representations for buttons and splitting poses and gestures into separate sections marked by headers.
- Added a way to compare poses and gestures by name.

- Added a utility for events to score based on specific poses and gestures: a matching poses return the score value, while a non-matching one returns zero.
- Unified implementation for “Gesture” initializers.

While we considered creating a null case for poses to aid in creating gestures that aren’t dependent on a pose and to let developers have an idle stance available for use, we decided against it for now based on the following reasons:

- Each motion-controlled game has it’s own definition of what an idle state actually means, usually meaning that it didn’t identify any other pose.
- Creating Pose Events with null makes it so the gesture recognize always return a new pose event after every placement registered in the component, no matter the threshold values.
- Based on this, defining a reference score for a null pose requires further analysis about the recognition process, including developing an alternative approach for processing poses and gestures that is out of scope for this research.

In regards to the game’s design, we found out that there is a conflict of actions when raising a hand, as it both related to collecting an orb and being part of the gesture to accelerate the player. This approach led to players unintentionally being unable to collect orbs raised above their heads without raising the player’s speed, creating unintentional behavior.

6.5 Chapter Conclusion

This section presented EdKit’s instances, which helped validate it’s features for the context of game development by identifying it’s strongest components and progressively evaluating the architecture after each iteration.

Changes in the gesture module mostly focused on removing duplicated code to make it easier for developers to code their own poses and gestures. On the other hand, the feedback module required the most amount of changes in order to give more control to programmers of how an affordance can be executed.

While the analytics module didn’t undergo any big architectural changes, the development process allowed us to fix some pain points to make the API easier to use and to avoid error-proneness from programmers using the hot spots and improve the expressiveness of provenance primitives.

Even after applying EdKit to the instances, we still note that there’s room for improvement in order to make the package easier to use: while we already mentioned about

removing the dependency that gestures have on poses, Section 8.3 will also talk about future proposals to improve the package based on the data acquired in the evaluation process described in Chapter 7.

7 EVALUATION

In this section, we're going to present the evaluation steps performed in testing EdKit with developers: Section 7.1 analyzes the profile of the participants; Section 7.2 presents the materials and methods utilized; Section 7.3 talks about the procedure; Section 7.4 describes the pilot test to improve the experiment; Section 7.5 details the results of the experiment; Section 7.6 discusses the results; and Section 7.7 concludes by talking about the lessons learned and validity threats for the experiment.

As described in Chapter 4, we performed an experiment with game programmers with the goal of evaluating the usability of EdKit's API, determining the friction of adoption with developers who are using it for the first time.

These goals were chosen in order to focus the evaluation process in validating the main pillars of EdKit as a framework and it's usefulness for developers.

7.1 Participants' Profile

The target audience for this experiment was game developers with experience programming games and with knowledge in the Unity game engine. The participants being chosen through convenience: in total, 4 people took part in the final experiment. Optionally, we also looked for developers with prior VR experience in order to get as close as possible to our intended use for the package. From now, they will be referred as participants A, B, C and D.

Participants A and B were recruited at "Fortaleza Indie Jogos (FIND Jogos)", a local game development event where developers gather to exchange experiences and meet new contacts. They agreed on the spot to participate in the in-person version of the experiment, which was also conducted at the same location. The venue was chosen as it presented the highest odds of finding game programmers with Unity knowledge that could participate in the experiment. Participants C and D, on the other hand, were chosen through prior contacts at a research group and were invited to participate in the remote version of the experiment, which was conducted through Google Meet.

Table 5 paints a picture of their general profile as game programmers, which we will dive deeper into below. All participants had, at most, 2 years of experience with Game Programming: participants either had beginner-level experience (<1 year) or were junior developers (1-2 years).

Participant	A	B	C	D
Game Programming	<1 year	<1 year	1-2 years	1-2 years
Unity	<1 year	<1 year	1-2 years	1-2 years
Unreal	<1 year	None	None	None
Godot	None	<1 year	None	None
Custom-Made Engine	None	None	None	None
Other Engines	None	None	1-2 years	5+ years
C#	None	<1 year	1-2 years	1-2 years
C++	<1 year	None	3-5 years	<1 year
GDScript	None	<1 year	None	None
C	None	None	1-2 years	None
Other Languages	5+ years	<1 year	3-5 years	5+ years
VR Development	None	None	1-2 years	1-2 years

Table 5 – Participants' Programming Profile

However, some participants' answers highlighted having more programming experience than that, suggesting that some of the participants had prior experience with general application and traditional programming beyond just their game development knowledge.

All participants had no prior experience creating their own game engine but had complimentary experience in another game engine besides Unity. Highlighting these complimentary skills is important as some of the experience acquired with other game engines and programming languages can be transferrable to programming games in Unity with the C# language.

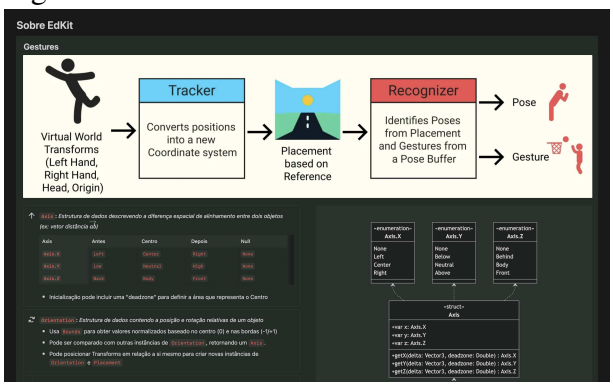
7.2 Materials and Methods

To conduct the experiment, we had four main artifacts used in the process: a branched-off, modified version of the "King" instance project; a digital form to collect data about the experiment and serve as a guide; a reference sheet containing details about EdKit and the tasks to be performed; Outside of the sample project, which had it's code written in English, all materials were written in Portuguese.

7.2.1 Sample Project

The sample project is a modified version of "King", in which we added elements that are related to the new pose that participants need to create in the experiment, but unrelated to functionality in EdKit (e.g. user interface elements, audio sources and more). After the test pilot, we also decided to integrate event hooks and wire events in order to streamline the experiment to focus only in implementing functionality from the package. The sample project contains as

Figure 21 – Reference Sheet



Source: Author

dependencies[1]: an evaluation version of the EdKit library, Unity SDKs for Virtual Reality (XR Interaction Toolkit); and Meta Voice SDK for Text-to-Speech capabilities.

7.2.2 Form Guide

Participants were guided through the process with a questionnaire (Appendix B) containing all stages of the experiment, as defined in Section 7.3. The guide form starts with a consent term to contextualize participants about how the experiment works and it's guidelines: participants were able to agree to the terms by entering their name and advancing to the next sections of the form:

– "Experience with Game Programming", containing all the questions from Activity 1.
– "Building a game using EdKit", containing all the questions for Activities 2 and 3.
– 'Post-Experiment', containing all questions for Activity 4.

The goal with this form was to simplify the logistics for conducting the experiment, as the participation in the experiment would only be validated and the results saved by completing it, with participants being able to leave at any point by just opting to not continue the form.

7.2.3 Reference Sheet

To aid participants in the experiment, we created a reference sheet containing information about how the sample project works and details about all EdKit's components: Appendix C contains a print version of the tutorial.

Figure 21 shows some example snapshots of the reference sheet. The document contains an introduction showing the goal of the tutorial, links to the sample project and form, as

[1] https://drive.google.com/file/d/1-t2cSpGR0nVVW4wpRmqekfgU_o_hPLU2/view?usp=sharing

well as recommendations for participants. The first section presents “King” with an overview of how the game works: it’s goals, controls and how to play the game, with a screenshot of the game.

The second section presents all of EdKit’s modules, with each module being structured as follows:

- Each important component gets a card detailing what it is, how it works and some important details about implementation, highlighting methods and types from the source code.
- Each module has a custom diagram showing the interaction between elements.
- For some items, an additional card is added showing a step-by-step of how the component can be added into EdKit.

The third section presents technical details about “King” that are relevant to the experiment, such as the command system, the components related to the player and a list of the poses implemented in and the rules behind them. Formatting is similar to the last section, but it is more condensed as it is intended to be a fast lookup reference where all of the content is visible in a single screen

The last section is dedicated to the step-by-step tutorial participants will perform with EdKit. It contains an initial summary of the new feature they must implement, followed by a set of tasks formatted in an hierarchical structure of what must be done based on module, task and actions.

7.3 Experiment

The procedure for this experiment was inspired by PONCIANO *et al.* (PONCIANO *et al.*, 2020) and aims to determine the usability of the API based on their former programming experience and how they perceived EdKit. The process was conducted under a single session and composed of four activities and it’s respective goals:

- 1. Programming Profile: establishing prior experiences with game programming.
- 2. Introduction to EdKit: informing about what is EdKit, it’s main features and a sample game made using the library.
- 3. Step-by-Step Tutorial: implementing a new feature in the presented sample game using all EdKit modules.
- 4. Post-Tutorial Questions: evaluating the usability and experience of EdKit and it’s API.

7.3.1 Programming Profile

In the first section, we asked participants about their profile around game programming, game engines, programming languages and VR development experience. All questions use an experience scale with four options (less than a year, 1-2 years, 3-5 years, 5+ years), plus an additional option for when the participant never had experience with the subject.

When talking about prior game engine experience, the options were selected as follows:

- The engines “Unity”, “Godot” and “Unreal” were chosen based on their popularity and presence in the game development community.
- “Custom Engines” were also added as an option to include the experience developers had previously building their own game engine.
- As for other engines not listed under any of the above categories, we list them under the option “Others”.

The options for programming languages were selected based on the main languages used for coding games in the selected game engines: “C#” for Unity and Godot; “C++” for Unreal; “GDScript” for Godot; “C” and “Others” for custom engines and programming languages that were not listed in this research.

7.3.2 Introduction to EdKit

After finishing the initial questionnaire, participants then are guided by the conductor through a brief introduction to what they’ll be doing in the experiment. The conductor presents the reference sheet and the sample project, explaining the game’s rules, it’s main components and how EdKit is used to implement in-game analytics, gesture detection and multimodal feedback.

Afterwards, participants were able ask questions about the explanation in order to have a clear understanding of the study artifacts before continuing on.

7.3.3 Step-by-Step Tutorial

In this stage, developers received a reference sheet containing information about EdKit and the sample project, as well as design specifications for a new feature they’re going to implement, with a step-by-step tutorial guiding on how this feature can be implemented. The participant had access to a computer with the following tools installed in order to conduct the

experiment (with the respective version in brackets):

- Unity Engine (2022.3.48f1)
- Visual Studio Code (1.93.1) with the Unity Extension (1.0.4) installed
- .NET (8.0.402)

The feature participants had to implement was a new command called “Block”, in which the player must put it’s left hand to the right and it’s right hand to the left, with the specifications being listed on the reference sheet. We chose this feature for the following reasons: it is thematic to the development of LRD skills, as it requires adding a crossed command for the developer to implement; it makes use of all of EdKit’s modules; and it is simple enough that can be implemented in a single session of 10 to 30 minutes, with time varying.

The reference sheet asks developers to complete tasks in the following order to implement the feature:

- Task 1: Implement a new “Block” command: with this task, developers use elements from the Gestures module (Axis, Orientation, Placement and Pose) to create a new command and learn about how the gesture system works.
- Task 2: Register the “Block” command to the list of identifiable commands: with this task, developers get a glimpse of the inner works of the command system and how a pose can be registered into EdKit.
- Task 3: Create a new Affordance to represent the “Block” command: with this task, developers learn how to create and configure an Affordance.
- Task 4: Connect the “Block” command and it’s affordance: with this task, developers learn how Affordances can be executed directly in the Editor.
- Task 5: Create a new Activity “Block” for the “Block” command: with this task, developers learn how to integrate EdKit’s analytics components in a game.
- Task 6: Registered the new “Block” activity in the player’s logs: with this task, developers learn how to register analytics information in a session.

While participants were allowed to ask questions to the conductor about Unity, EdKit and the sample game, any questions that gave away solutions for the tasks were not allowed. Furthermore, participants had access to compiler errors and log messages provided by the tools used to inform their process, but the participant was responsible for deciding when they had completed a task or not, with the conductor not being allowed to say or indicate that an activity was done correctly or not.

After considering the activity finished or at the end of the time dedicated for this activity, participants were prompted to indicate in a checklist tasks they think they completed and what types of advice did they ask the conductor. Participants also were asked about their level of understanding about EdKit, sample game and features they had to implement, which they had to respond using a Likert scale in five levels: “1 - Completely Disagree”, “2 - Disagree”, “3 - Indifferent”, “4 - Agree” and “5 - Completely Agree”.

7.3.4 Post-Tutorial Questions

To cap things off, participants answered questions related to their experience with EdKit after the tutorial. The questions on this part of the questionnaire were based on Cognitive Dimensions of Notations (BLACKWELL *et al.*, 2001) and they answered these questions using a Likert scale in five levels: “1 - Completely Disagree”, “2 - Disagree”, “3 - Indifferent”, “4 - Agree” and “5 - Completely Agree”.

Afterwards, participants evaluated EdKit based on the advantages and disadvantages before giving a final grade of their experience with the package with a linear scale going from 1 (“Very Bad”) to 10 (“Very Good”). We also left open a space for participants to voice their suggestions and comments about EdKit, gathering relevant feedback to inform future works.

7.3.5 Formats

This experiment was conducted into both in-person and remote formats, with both using the same procedure for all participants. Table 6 shows time estimates for each format.

Activity	In-Person	Remote
Programming Profile	5 mins	5 mins
Introduction to EdKit	10 mins	15 mins
Step-by-Step Tutorial	45 mins	90 mins
Post-Tutorial Questions	5 mins	10 mins
Total	65 mins	120 mins

Table 6 – Time estimates for Evaluation

For the in-person experiment, we conducted it in a session of 65 minutes, with the amount of time chosen based on venue limitations and with the goal of gathering as many participants as possible. Participants A and B took part in this format, which used a computer provided by the conductor containing the required tools and already had multiple instances of

the project sample configured for use.

For the remote experiment, we conducted it in a session of 120 minutes: while it wasn't limited by any venue restrictions, the extension of time was made mostly in order to accommodate for possible time risks involving Unity's domain reload. Participants C and D took part in this format, using their own computers and sharing their screen during the Introduction to EdKit and Step-by-Step tutorial activities in order for the conductor to follow along.

7.4 Test Pilot

To improve this experiment before conducting it with participants, a test pilot was conducted in order to inform on the quality of the experiment and adjust it to make it doable under time constraints. To perform this pilot test, a member of the research team was recruited to participate: from now on, they will be referred to as participant T.

The test pilot was performed with the following goals in mind:

- Determine the estimated duration of the experiment and how it could be improved to fit in the target duration.
- Evaluate the potential of applying the experiment asynchronously with a guide
- Evaluate the quality of the questions elaborated

Participant T is a junior game programmer with less than a year of experience with VR development, Unity, C, C# and other game engines, while also having 1-2 years of experience in Godot Engine, GDScript and other programming languages.

Overall, participant T was pretty positive about the package and had a clear understanding of EdKit, the sample project and the new feature being worked on. The participant was able to correctly state and perform all tasks, requiring help for Task 1 and to understand better the requirements for the new feature, how "EdKit" and the sample "King" worked, as well as general Unity tips.

In terms of API, participant T agreed that EdKit helped him develop games in Unity, making the design and implementation easier, being a powerful package that reduces errors, makes code more legible and fulfill with the required specifications. At the same time, they were also unsure about being able to fulfill the activity with EdKit, nor agreeing or disagreeing about the package being more restrictive or harder to use and agreeing that it didn't have everything they needed.

While the participant was able to complete all of the tasks successfully, the exper-

iment took over 6 hours. This was due to issues related to Unity's domain reload, which is when Unity recompiles all of it's assemblies after a change in code for use in the Editor. After each code change related to a task in the participant's machine, Unity would freeze for up to 10 minutes during the domain reloading operation, presenting a challenging environment for an evaluation.

Another problem encountered was with the explanations available, which were insufficient as the participant asked the conductor frequently about features not discussed in the introduction to EdKit, as well as confusion related to poor wording and unclear instructions on the tutorial.

Some actions were taken in order to keep the length of the experiment under control and viable for applying it asynchronously:

- The step-by-step tutorial was reduced to focus mostly on implementing the "Block" command using EdKit, removing unnecessary extra steps that involved the Unity-specific concepts (e.g. registering Unity Events in the Editor that are unrelated to the activity at hand). The tasks were also adjusted to not rely on Unity's Domain Reload as much while preserving the essence of implementing code using EdKit. We also reviewed sentences to make the tutorial clearer and more understandable while not giving away the solution. A time limit was also added to Activity 3 in order to keep it on schedule and not take too much of participants time or create scenarios where participants gave up on the experiment midway through.
- For evaluating with developers at an in-person event, a machine with a solid state drive was prepared with the necessary tools and instances of the sample project preloaded in to mitigate the time spent reloading the project during the experiment.
- Questions were left unchanged, but the process of answering them was changed to not rely on the conductor or the machine used for the experiment. With this, the conductor is able to have up to three participants performing the experiment in parallel: one answering the questionnaire for Activity 1, guiding another through Activities 2 and 3, and another finishing the questionnaire by itself.
- The introduction of EdKit was overhauled with new resources added to the README of the project on Github, including sketches showing features of EdKit and the sample game not yet present on the presentation. This information can now be used as reference material during Activity 3 to make participants less reliant on asking the conductor for tips

Participant	A	B	C	D
EdKit	Indifferent	Agree	Completely Agree	Completely Agree
“King” Game	Agree	Completely Agree	Completely Agree	Completely Agree
New Feature	Agree	Indifferent	Agree	Agree

Table 7 – Participants’ opinions on their understanding of the tutorial

Participant	A	B	C	D
Task 1	Yes	Yes	Yes	No
Task 2	No	Yes	No	Yes
Task 3	Yes	Yes	No	No
Task 4	No	No	No	No
Task 5	No	No	No	Yes
Task 6	No	No	Yes	No
New Feature	No	No	No	Yes
EdKit	No	Yes	Yes	Yes
“King”	No	Yes	No	No
Unity	No	No	No	Yes

Table 8 – Participants’ Help Requests

during this stage, pushing them back to their main role of keeping the experiment going smoothly.

7.5 Results

For this experiment, we observed the participants’ responses to the experiment, how their profile and perception stacked against their performance in the step-by-step tutorial and their impressions about EdKit, including suggestions and improvements presented.

7.5.1 Introduction to EdKit

Tables 7 and 8 shows how participants interpreted the concepts presented by the conductor after having hands-on experience and the moments when they requested help from the conductor, respectively.

Overall, EdKit and the requirements for the new feature were well received: while participants showed confidence in their comprehension of the package, not all of them completely agreed with the tutorial. Participant B was more neutral about understanding the concepts surrounding EdKit, diverging a bit from other participants in that regard. They were also the participant with the most divergence between their perceived results and actual performance.

All participants agreed to having clear understanding of how the sample game worked, while also all requesting tips from the conductor, with the most help requested being

about how EdKit works and for Tasks 1 and 3.

7.5.2 Step-by-Step Tutorial

To evaluate the participants' tasks in Activity 3, their performance for each task was categorized in one of three categories:

- **Success**: Participant completed the task successfully following specification requirements
- **Partially Complete**: Participant completed the task successfully, but didn't follow the specification requirements
- **Failed**: Participant wasn't able to complete the task

This distinction between "Success" and "Partially Complete" was noted by the conductor to not let cases where the command was implemented by the programmer, but due to a lack of time and/or comprehension didn't follow the specifications for the new command.

Participant	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
A	Success	Failed	Success	Failed	Failed	Failed
B	Success	Success	Success	Success	Failed	Failed
C	Failed	Success	Success	Success	Success	Success
D	Success	Success	Success	Success	Success	Success

Table 9 – Participants' Perceived Performance

Participant	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
A	Success	Success	Partially Complete	Failed	Failed	Failed
B	Partially Complete	Failed	Partially Complete	Partially Complete	Partially Complete	Failed
C	Partially Complete	Success	Success	Partially Complete	Partially Complete	Failed
D	Partially Complete	Partially Complete	Success	Partially Complete	Success	Success

Table 10 – Participants' Actual Performance

Tables 9 and 10 shows the tasks participants think they performed and results noted by the conductor during the experiment respectively. While participants B, C and D were able to complete their efforts within the time limit, participant A only reached the third task before time ran out, leaving tasks 4-6 untouched.

In task 1, participants encountered issues related to naming conventions in the project, being unsure of where the poses were located and what purpose some classes served in the

experiment. While participants in the in-person experiment had access to code completion features, the feature wasn't used by any of the participants to look for useful methods in the package.

These were the main sources of tips previously requested to the conductor, showing an obstacle that was embedded into the experiment. In the end, all participants ended up opting to create the new pose instancing a class instead of creating a type and conforming it to the respective pose interface.

Pain Points for Task 1: (i) Creation of the "Pose" instance as a static variable; (ii) Where to implement the pose in the project; (iii) Creating a new "Axis" instance based on two orientations and using it's enumerators to obtain an outcome.

Task 2 was easier in comparison, as participants were faster to decide where to place their implementation in the files. However, we still saw some issues where some participants registered the command for recognition, but not for selection. Participant B also wasn't able to register the command, even though they considered it done when responding to the poll, being also the only participant to ask for help in this section.

Pain Points for Task 2: (i) Adding the new command to the list of recognizable commands; (ii) Following the specification of the tutorial

Task 3 was considered the easiest task of the experiment, with everyone being able to complete it quickly and adjusting parameters in the Editor without many issues. However, Participants A and B asked for tips in this part and were unable to follow the specifications for what representations to use in the Affordance.

Pain Points for Task 3: (i) Following the specification of the tutorial

Task 4 also had similar results, where all participants who reached this stage of the experiment being able to complete it, with only participant D needing some help. However, participants weren't able to nail the specification requirements: while some were able to apply the "Block Request" affordance on the right object, the tutorial asked participants to fade other user interface components related to other gestures when the request command was highlighted.

Pain Points for Task 4: (i) Locating the object related to the user interface element for the "Block" command; (ii) Understanding what is the role of "CommandRequester" in the project; (iii) Wiring the events with the specified affordance requirements

Task 5 was perceived as the hardest task in the experiment: while participants didn't state on the record that they asked for help with this task, they struggled with what they needed to do in order to call the event for when a block is performed. Participants asked about aspects of the package and the sample project, as well as where to place the logic and what the "NextCommand" property in the command selector was for.

Only participant D stated that they had requested help about this task in specific, as other participants treated the request as clarifications about EdKit and the sample project.

Pain Points for Task 5: (i) Defining the attributes for the activity for the "Block" command; (ii) Defining the necessary parameters for the analytics methods; (iii) Modifying the method "EvaluateLastPose" method to fire the "onBlockPerformed" event correctly;

While Task 6 was easier compared to the last task, people had difficulty understanding how to use the event for when the block is performed to register events into the "SessionBehaviour" component using the implementation built previously. Only participant C stated that they asked for help in this part of the activity.

Pain Points for Task 6: (i) Wiring the required method to the "OnBlock" event;

7.5.3 Post-Tutorial Questions

To cap things off, participants also were asked about their impressions about EdKit and it's API usability.

Table 11 shows the overall impressions of participants, with some highlights being listed below:

- All participants agree that EdKit is a package that makes things easier to implement and produces more legible code.
- Participants were divided about EdKit making the design easier and having everything they need to fulfill the requirements.
- All participants, except A, agreed that EdKit is a powerful framework.

Participant	A	B	C	D
“EdKit’s syntax helped you develop games in Unity?”	Agree	Indifferent	Agree	Completely Agree
“I had to give up on implementing the features because I didn’t know if I would be able to implement them using EdKit”	Agree	Completely Agree	Disagree	Completely Disagree
“Does EdKit make the design easier?”	Indifferent	Disagree	Agree	Completely Agree
“Does EdKit make implementation easier?”	Agree	Agree	Agree	Agree
“Is EdKit hard to use?”	Indifferent	Agree	Agree	Completely Disagree
“Is EdKit powerful?”	Indifferent	Agree	Agree	Completely Agree
“Does EdKit have all the features that I need?”	Indifferent	Indifferent	Agree	Completely Disagree
“Does EdKit restrict my freedom as a programmer?”	Completely Disagree	Disagree	Agree	Completely Disagree
“Is the code more legible with EdKit?”	Agree	Agree	Agree	Completely Agree
“Does less errors occur with EdKit?”	Indifferent	Indifferent	Agree	Completely Agree
“Does EdKit fulfill better with the requested specifications?”	Agree	Indifferent	Agree	Completely Agree
Final Rating for EdKit	7	7	9	10

Table 11 – Participants’ Impressions of EdKit

- All participants, except B, agreed on EdKit better fitting required specifications and the syntax helping develop games in Unity.
- Only participant C found EdKit restrictive for use.
- Only participant D didn’t find EdKit hard to use.
- Participants C and D consider that less errors occurred with EdKit, while A and B were unsure about it, to the point they state having to give up implementing parts of the process.

7.6 Discussion

While there’s still challenges to make EdKit easier to use and offer more resources to reduce barrier of entry and ease the learning process, reception for the package was pretty positive between participants. In general, participants with more knowledge of C# and Unity ended up performing better in the experiment, as well as their use of EdKit.

7.6.1 Advantages

Advantages (i) Perceived by developers as powerful, legible and capable of fulfilling it's main requirements; (ii) Data-driven declarative design for poses, gestures and affordances helped make the concepts easier to understand; (iii) More accessible code; (iv) Reduced development time.

The experiment showed that there is potential for improving development times and making the process easier for creating games with LRD concepts using EdKit, with developers having a pretty positive reception for the package. Participants found that the package was easier to use and made code more legible, with no opinions about it being not powerful enough, leading to more errors or not fulfilling the requirements asked.

When looking after the strongest points for the package, we can see that creating poses and affordances were the only activities where all participants were successful, even if only partially. While the Gesture module is based on a declarative design approach that is very much focused on scripting, the Feedback module focus on direct customization by assigning references in the Editor.

The use of a declarative data-driven approach made EdKit more legible and easier to understand, creating reusable components that can be expanded upon and composed together in different ways, making code more accessible and reducing development time. This was also highlighted by the participants themselves:

- A found the methods and classes intuitive and easier to understand.
- B notes that EdKit helps reduce the overall development time of these features.
- C considers EdKit a great way to make code more accessible.
- D highlights how EdKit offers intuitive abstractions, but also highlights how the package makes it easier for converting positions for the hands and head.

7.6.2 Disadvantages

Disadvantages (i) Presence of confusing hot spots in the API; (ii) Data-driven declarative design makes it harder for integration; (iii) Creating and registering Analytics isn't a clear process.

While a lot of praise was given in terms of syntax, legibility, easier implementation and design, participants found the package hard to use during their hands-on time. Besides the previously mentioned feedback of improving the learning resources available for the framework,

we also consider that there's still some pain points in the API that can be addressed for a better overall experience with EdKit. Some examples of this include:

- In "Axis", participants were confused about the return type of properties such as "x", "y" and "z": they expected a numerical value to be returned, but instead they got an enumerator that changes for each property ("Axis.X" for "x", "Axis.Y" for "y" and "Axis.Z" for "z"). This also presented another challenge to participants, as they were unsure of how exactly to compare these values to return a numerical value required by the scoring algorithm of a "Pose". While EdKit has a utility method called "Global.ScoreComparisons" to compose a score out of boolean comparisons using the enumerators, participants either asked for help at this stage or looked for other code examples in the sample project.
- "PoseData", while a non-scripting solution for poses that was accessible to create in the Editor, didn't end up becoming part of the final experiment due to the pilot test encountering fundamental issues with it's usability and how hard it was to configure the object. The type is currently deprecated and requires a rework of it's user experience to be more friendly to users, but future works show that this current pain point can be a great way to create poses and gestures.
- Some abstractions in the "Gesture" module and the scoring algorithm for poses uses a declarative approach for implementation.

While participants were able to create the poses successfully, the same cannot be said of integrating them into the sample project, as participants encountered challenges to add the new command into the recognition system.

The same also happened to the Analytics module, in which only participant D was able to successfully register provenance primitives in the session in Task 6. Combined with the fact that Task 5 was the hardest task for participants, this displays how the module is hard to understand in comparison to the other two, as many doubts arose both the experiment requirements and integrating the API with other components and modules.

7.7 Chapter Conclusion

In this chapter, we presented the first evaluation of EdKit carried out with 5 Unity developers, which provided important insights about this research. Below, we'll describe the lessons learned and possible threats to the research.

7.7.1 Lessons Learned

Takeaways (i) Provide better learning resources for EdKit; (ii) EdKit is better understood by advanced developers; (iii) More studies are required to validate the research's findings.

While EdKit was well documented and participant D praised it's depth in explaining the concepts and how well organized the project was, we consider that lack of documentation around the sample project may have impacted the experiment negatively, as expressed by participants' A and C suggestions about further improvements.

Participants had challenges locating files and methods in the sample project, with Participant C highlighting how they were unsure of where to place the code during one of the tasks, suggesting the addition of comments indicating where the implemented code should stay. Some were also interested in a diagram showing a layout of how the project is organized.

Overall, participants wanted improved documentation resources, which include more code examples, learning resources, references and sample projects with source code being available for reference. This is reinforced by their behavior in the experiment, as they used both the reference sheet and the sample project's source code to perform the tasks.

Another conclusion we reached is that EdKit is not a package that is friendly to game programmers with beginner-level experience of Unity and C#: besides it's documentation and sample projects, the package doesn't offer a helper utility to aid developers in getting started using EdKit's features in their projects. Participants also showed struggles with understanding elements related to Unity and C#, but were not directly part of the framework, with one example of this being how Unity Events could be leveraged to connect components and build functionality while avoiding more direct dependencies.

EdKit and the sample project used in the evaluation makes use of essential Unity concepts (e.g. MonoBehaviours, Scriptable Objects, Unity Events) and more intermediate C# concepts (e.g. lambda functions, extensions and delegates), making for a challenging experience for beginner developers just starting their journeys in any of these two areas. Task 3 showed us that offering non-scripting solutions can reduce the barrier of entry and improve the ease of use for EdKit can make some of these resources more accessible.

We consider that the package should maintain it's core pillars of development, requiring a minimum level of programming experience in Unity and C# for more effective use. However, we should also look into providing non-scripting accessible offerings that can make

features of the package usable by beginners and developers who are not programmers, at the cost of limited functionality.

At the same time, the level of experience wasn't evenly split between experiment formats, with only beginners in Unity and C# taking part in the in-person format and more knowledgeable programmers participating on the remote one. Not only that, but weren't able to recruit senior programmers to participate in the experiment, as well as junior developers for the in-person format. Therefore, we cannot conclusively say what impact the experiment format had in their performance: while this will be discussed more in-depth in Sections 7.7.2 and 8.3, we consider that this can be resolved by more in-depth evaluations of the package with game developers. Future evaluations can also use a different sample project in use, helping discern between what is an issue with EdKit, what is an issue with the sample being used and what is an issue with how EdKit and the sample project's code are integrated.

7.7.2 Threats to Validity

This section presents threats to the validity of this research, showing how we aimed to mitigate them given the time and scope constraints and how they can be tackled on future works.

In the in-person format, the keyboard layout in the operational system didn't fully match the physical keys of the computer provided, affecting some control keys and symbols (e.g. ";", "[", "]"). Since this was encountered mid-experiment, we weren't able to fix it in time: while in-person participants stated about having little annoyances, none of them said that it was harmful to their experience, even though it affected the amount of time they had available.

Expanding on this, another possible threat is that the time was not enough to fully evaluate EdKit as a package, even in the remote format.To mitigate this, we conducted a pilot test to make the experiment doable within the estimate time constraints for this research and that informed us to rework the experiment into making it more focused on the use of EdKit-specific workflows. We also looked into making the experiment formats take an equivalent amount of time by taking into account limitations we found about Unity's domain reload for remote participants, as well as time and venue restrictions for the in-person participants.

Another threat is that the procedure used is not representative of an actual development scenario where EdKit would be used. To make the procedure more representative, we built a new sample project using the "King" instance for the evaluation and written a development

scenario in which participants had to create a new feature with a written set of requirements.

While we already were able to evaluate EdKit with both a remote and in-person settings, which already presented changes relating to devices, Unity versions and other configurations, the environment was still very much controlled. All participants in the experiment had a conductor and learning resources to ease the use of EdKit in a package, with guidance that is not representative of actual development scenarios. Therefore, testing the package in other environments (e.g. not having a conductor, starting a project from scratch, using a different version of Unity, etc.) can help identify how EdKit's impressions extend beyond a guided and controlled environment and what actions can be done to make EdKit easier to integrate in a project.

We also considered that, due to the specific-ness of the topic, that the research may not be relevant enough for it to be studied as a standalone topic. To mitigate this, we produced two papers to evaluate the research topic about VR games for PVI working with LRD concepts and how EdKit can aid game developers in making games with this topic, respectively. More details about this will be available in Section 8.2.

8 FINAL CONSIDERATIONS

We expect EdKit to become a useful tool that can be added into a toolset of game developers working with OM games, especially for makers creating experiences with LRD concepts in an VR environment in Unity.

This chapter concludes the dissertation and presents the following structure: first, in Section 8.1 we'll return to the main research question and answer it based on the knowledge presented; then, Section 8.2 will present the contributions made by this research; finally, Section 8.3 talk about future avenues to further expand this work.

8.1 Main Research Question

This dissertation focused on answering the question of "How can we bring Left-Right Discrimination concepts for the development of Virtual Reality games for People with Visual Impairments?", with the goal of simplifying the integration of these concepts in games and the hypothesis that a dedicated artifact would make it easier for developers to make them.

Based on the systematic mapping presented in Chapter 2, we saw that some of the main approaches for using LRD concepts are through game design, the use of spatial audio techniques and multimodal feedback. By taking into account each vision profile and the toolset of techniques, developers can create experiences that more efficient, immersive and autonomous so PVI enjoy their time in a fun VR experience.

With the development of EdKit, we were able to bring some of these concepts to the VR development of games, such as:

- Plane Slicing, through the use of "Axis" enumerators
- Body Image, through the focus on poses and gestures based on the player's limbs and body orientation
- Depth Perception, through the use of Affordance Effects
- Clue Recognition, through the use of Affordances providing abstract clues
- Spatial References, through the use of "Orientation" and "Placement" instances, as well as egocentric system for poses and references

Furthermore, Motor Coordination, Dominant Side and Aligned and Crossed Commands did also make in into the package, but only implicitly through the pose and gesture systems, as more development is required to make these concepts explicit in it's architecture.

We designed EdKit to be modular, reusable and able to be extended by other developers, which could create new poses and gestures, implement new feedback systems or expand the analytics system. With this, we hope the package can be used in any game made in Unity that works with OM and LRD concepts in an explicit or implicit manner. To the best of our knowledge, this is the first work designed as an open-source Unity package that offers a tool for developers to embed LRD concepts into their applications.

8.2 Contributions

This work presented the following contributions, which are listed below:

- A state of the art in how VR games for PVI work with LRD concepts as described in the systematic mapping presented in Chapter 2, highlighting how games for PVI are using LRD concepts; examples of other VR applications for PVI and the techniques created to help PVI in a virtual environment; common features used in games for PVI; observed effects, challenges and limitations in improving their experience in VR and their skills.
- EdKit, a custom Unity package to aid in the development of games working with LRD concepts, as well as the design process behind it's design decisions.
- Sample Unity project containing three small VR games demonstrating how to use EdKit in a project to work with LRD concepts.

In terms of achieved results, the evaluation showed that EdKit already has good pillars for development, with features relevant to developers that make things easier to use, with further development only strengthening the feature set on the package. The sample project served both as a proof-of-concept for how EdKit can be used and contributing games that can be played by PVI.

8.2.1 Publications

Beyond the contributions listed above, there were also publications of papers in conferences about the research and the project which originated it, which are listed below:

- *Uma ferramenta para a customização de ambientes virtuais para práticas de Orientação e Mobilidade* (JÚNIOR *et al.*, 2022). Accepted as a short paper in WebMedia 2022, under the "Workshop de Ferramentas e Aplicações (WFA)" track and receiving a "Best Work" award. Introduces the E3 tool to create indoor maps using a web editor and the ENA player,

responsible for rendering these maps as tridimensional virtual environments that PVI can interact with. Authors: José Martônio L. de M. Júnior, Agebson R. Façanha, Maria da C. Carneiro Araújo, Jaime Sánchez, Windson Viana

– *Dashboards to support rehabilitation in orientation and mobility for people who are blind* (MORAES *et al.*, 2023). Accepted as a short paper in WebMedia 2023 under it's main track. Expands on the work previously cited by adding in the integration of game analytics and customization methods for OM teaching of PVI. Authors: José Martônio L. de M. Júnior, Phyllipe Do Carmo F., Maria da C. Carneiro Araújo, Bernardo Costa, Myrna Amorim, Agebson R. Façanha, Joel dos Santos, Windson Viana
– *Empowering Orientation and Mobility Instructors: Digital Tools for Enhancing Navigation Skills in People Who Are Blind* (FAÇANHA *et al.*, 2024). Accepted as a conference paper on International Conference on Computers Helping People with Special Needs (ICCHP) 2024 as a Special Thematic Session. It continues on the work done with OM indoor virtual environments, with feedback obtained after an evaluation with HCI experts, OM instructors and participants. Authors: Agebson R. Façanha, José Martônio L. de M. Júnior, Maria da C. Carneiro Araújo, Joel dos Santos, Jaime Sánchez, Windson Viana
– *Enhancing Orientation and Mobility Skills for Persons with Visual Impairments through VR Games: Introducing the EdKit Unity Package*. Accepted on International Conference on Entertainment Computing (ICEC) 2024 under the "Works in Progress" track, in which it received a "Best Paper" award. Contains the initial concepts behind EdKit's design and was adapted into Chapters 4 and 5, which were further developed with more details as development of EdKit continued on. Authors: José M. L. de Moraes Júnior, Agebson R. Façanha, Bruno C. da Silva, Windson Viana, Joel dos Santos
– *Virtual Reality for People with Visual Impairments in the development of Left-Right Discrimination Skills: A Systematic Mapping Study* (JÚNIOR *et al.*, 2024). Accepted as a full paper on WebMedia 2024 under the "I Workshop de Revisões Sistemáticas de Literatura em Sistemas Multimídias e Web" track. Talks about the systematic mapping done in Chapter 2 and was later adapted into Chapters 1 and 2. Authors: José M. L. de Moraes Júnior, Agebson R. Façanha, Windson Viana

8.3 Future Work

While this research focused on VR games for PVI with LRD concepts, it's important to widen the scope of the theme and explore related pathways. The goal with this is to fill in the gaps found by the systematic mapping in Chapter 2 by exploring further on how LRD concepts are used and developed by players in digital games: this theme aims to identify common idioms that can be brought over to VR games for PVI. While the systematic mapping already captured some examples (GONÇALVES *et al.*, 2023a), further literature reviews can help find game examples with more explicit design techniques and provide case studies for the area.

In future work, EdKit can also be evolved in the following directions:

- Further abstract the package's core logic to become an engine-agnostic C# framework with engine-specific adapters.
- Specialize EdKit to use more Unity features (e.g. integrating poses and gestures with Unity's New Input System).
- Branch each module into their own package to be reused in other contexts (BOYD, 1993).
- Integrate implicit LRD concepts more explicitly in the modules of the framework.
- Implementing already existing and proven techniques focused on VR games for PVI using the package (e.g. PENUELA *et al.* (PENUELA *et al.*, 2022)).
- Expand the evaluation EdKit with more participants and different procedures.
- Apply the package into more VR games for PVI and in projects of different scopes.
- Test the games with PVI, looking for improvements and player-centric requirements.
- Develop custom tools for OM instructors that can use EdKit's analytics module.

REFERENCES

ANDRADE, R. *et al.* Echolocation as a means for people with visual impairment (PVI) to acquire spatial knowledge of virtual space. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, USA, v. 14, n. 1, p. 1–25, 2021.

AZIZ, N.; STOCKMAN, T.; STEWART, R. Planning your journey in audio: design and evaluation of auditory route overviews. **ACM Transactions on Accessible Computing**, ACM New York, NY, v. 15, n. 4, p. 1–48, 2022.

BLACKWELL, A. F. *et al.* Cognitive dimensions of notations: Design tools for cognitive technology. In: **Cognitive Technology: Instruments of Mind: 4th International Conference, CT 2001 Coventry, UK, August 6–9, 2001 Proceedings**. [*S. l.*]: Springer Berlin Heidelberg, 2001. p. 325–341.

BOSCH, J. **Object-Oriented Frameworks : Problems & Experiences**. [*S. l.*], 1997. (Blekinge Institute of Technology Research report, 9).

BOYD, N. Building object-oriented frameworks. **The Smalltalk Report**, v. 3, n. 1, p. 1–6, 1993.

BRULE, E. *et al.* Mapsense: multi-sensory interactive maps for children living with visual impairments. In: **Proceedings of the 2016 CHI conference on human factors in computing systems**. [*S. l.*: *s. n.*], 2016. p. 445–457.

BURTE, H.; MONTELLO, D. R. How sense-of-direction and learning intentionality relate to spatial knowledge acquisition in the environment. **Cognitive Research: Principles and Implications**, Springer, v. 2, p. 1–17, 2017.

CHANG, J. S.-K. *et al.* Keep the ball rolling: Designing game-based tangible VR for spatial penetrative thinking ability. In: **Proceedings of the 2019 on Designing Interactive Systems Conference**. [*S. l.*: *s. n.*], 2019. p. 215–226.

CHOUEIB, A.; BERKMAN, M. İ. Comparing Performance and Experience in VR vs. Real-World Through a Puzzle Game. In: **International Conference on Videogame Sciences and Arts**. [*S. l.*]: Springer Nature Switzerland, 2023. p. 72–85.

COVACI, A.; OLIVIER, A.-H.; MULTON, F. Third person view and guidance for more natural motor behaviour in immersive basketball playing. In: **Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology**. [*S. l.*: *s. n.*], 2014. p. 55–64.

DOVE, G. *et al.* Digital Technologies in Orientation and Mobility Instruction for People Who are Blind or Have Low Vision. **Proceedings of the ACM on Human-Computer Interaction**, ACM New York, NY, USA, v. 6, n. CSCW2, p. 1–25, 2022.

ELOR, A. *et al.* On shooting stars: Comparing cave and hmd immersive virtual reality exergaming for adults with mixed ability. **ACM Transactions on Computing for Healthcare**, ACM New York, NY, USA, v. 1, n. 4, p. 1–22, 2020.

FAÇANHA, A. R. *et al.* O&M indoor virtual environments for people who are blind: A systematic literature review. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, USA, v. 13, n. 2, p. 1–42, 2020.

FAÇANHA, A. R.; OTHERS, J. d.; SÁNCHEZ, J.; VIANA, W. Empowering orientation and mobility instructors: Digital tools for enhancing navigation skills in people who are blind. In: **International Conference on Computers Helping People with Special Needs**. [*S. l.*]: Springer Nature Switzerland, 2024. p. 436–443.

GONÇALVES, D. *et al.* "My Zelda Cane": Strategies used by blind players to play visual-centric digital games. In: **Proceedings of the 2023 CHI conference on human factors in computing systems**. [*S. l.*: *s. n.*], 2023. p. 1–15.

GONÇALVES, I. *et al.* Inclusive social virtual environments: Exploring the acceptability of different navigation and awareness techniques. In: **Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems**. [*S. l.*: *s. n.*], 2023. p. 1–7.

GORMLEY, G. J.; BRENNAN, C.; DEMPSTER, M. 'What... you can't tell left from right?'medical students' experiences in making laterality decisions. **Medical Education**, Wiley Online Library, v. 53, n. 5, p. 458–466, 2019.

GRIFFITHS, D.; BESTER, A.; COVENTRY, K. R. Space trumps time when talking about objects. **Cognitive Science**, Wiley Online Library, v. 43, n. 3, p. e12719, 2019.

GUARESE, R.; ZAMBETTA, F.; SCHYNDEL, R. van. Evaluating micro-guidance sonification methods in manual tasks for blind and visually impaired people. In: **Proceedings of the 34th Australian Conference on Human-Computer Interaction**. [*S. l.*: *s. n.*], 2022. p. 260–271.

GURP, J. V.; BOSCH, J. Design, implementation and evolution of object oriented frameworks: concepts and guidelines. **Software: Practice and Experience**, Wiley Online Library, v. 31, n. 3, p. 277–300, 2001.

HEGARTY, M. *et al.* Spatial abilities at different scales: Individual differences in aptitude-test performance and spatial-layout learning. **Intelligence**, Elsevier, v. 34, n. 2, p. 151–176, 2006.

HOOGSTEEN, K. M. *et al.* Beyond the cane: describing urban scenes to blind people for mobility tasks. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, v. 15, n. 3, p. 1–29, 2022.

INDIA, G.; JAIN, M.; KARYA, P.; DIWAKAR, N.; SWAMINATHAN, M. VStroll: An audio-based virtual exploration to encourage walking among people with vision impairments. In: **Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility**. [*S. l.*: *s. n.*], 2021. p. 1–13.

JACOBSEN, E. E.; KRISTENSEN, B. B.; NOWACK, P. Patterns in the analysis, design and implementation of frameworks. In: **Proceedings Twenty-First Annual International Computer Software and Applications Conference (COMPSAC'97)**. [*S. l.*]: IEEE, 1997. p. 344–348.

JR, J. J. L. A discussion of cybersickness in virtual environments. **ACM Sigchi Bulletin**, ACM New York, NY, USA, v. 32, n. 1, p. 47–56, 2000.

JÚNIOR, J. M. L. d. M. *et al.* Uma ferramenta para a customização de ambientes virtuais para práticas de orientação e mobilidade. In: **Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)**. [*S. l.*]: SBC, 2022. p. 103–106.

JÚNIOR, J. M. L. de M.; VIANA, W.; FAÇANHA, A. R. Virtual reality for people with visual impairments in the development of left-right discrimination skills: A systematic mapping study. In: **Anais Estendidos do XXX Simpósio Brasileiro de Sistemas Multimídia e Web**. [*S. l.*]: SBC, 2024. p. 185–197.

KARAOSMANOGLU, S. *et al.* Canoe VR: An immersive exergame to support cognitive and physical exercises of older adults. In: **CHI Conference on Human Factors in Computing Systems Extended Abstracts**. [*S. l.*: *s. n.*], 2022. p. 1–7.

KIM, S.; LEE, K.-p.; NAM, T.-J. Sonic-badminton: audio-augmented badminton game for blind people. In: **Proceedings of the 2016 CHI Conference extended abstracts on human factors in computing systems**. [*S. l.*: *s. n.*], 2016. p. 1922–1929.

KITCHENHAM, B. *et al.* Systematic literature reviews in software engineering–a systematic literature review. **Information and software technology**, Elsevier, v. 51, n. 1, p. 7–15, 2009.

KITCHIN, R. M.; JACOBSON, R. D. Techniques to collect and analyze the cognitive map knowledge of persons with visual impairment or blindness: Issues of validity. **Journal of Visual Impairment & Blindness**, SAGE Publications Sage CA: Los Angeles, CA, v. 91, n. 4, p. 360–376, 1997.

KOKKINARA, E.; SLATER, M.; LÓPEZ-MOLINER, J. The effects of visuomotor calibration to the perceived space and body, through embodiment in immersive virtual reality. **ACM Transactions on Applied Perception (TAP)**, ACM New York, NY, USA, v. 13, n. 1, p. 1–22, 2015.

KREIMEIER, J.; GÖTZELMANN, T. FeelVR: Haptic exploration of virtual objects. In: **Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference**. New York, NY, USA: Association for Computing Machinery, 2018. p. 122–125.

KREIMEIER, J.; GÖTZELMANN, T. Two decades of touchable and walkable virtual reality for blind and visually impaired people: A high-level taxonomy. **Multimodal Technologies and interaction**, MDPI, v. 4, n. 4, p. 79, 2020.

LABS, O. **Owlchemy Labs Shares a First of It's Kind Vision Accessibility Update for Cosmonious High**. 2023. Disponível em: https://owlchemylabs.com/blog/owlchemy-labs-shares-the-first-of-its-kind-vision-accessibility-update-for-cosmonious-high. Acesso em: 25 ago. 2024.

LAHAV, O. Virtual reality systems as an orientation aid for people who are blind to acquire new spatial information. **Sensors**, MDPI, v. 22, n. 4, p. 1307, 2022.

LAHAV, O.; SCHLOERB, D. W.; SRINIVASAN, M. A. Newly blind persons using virtual environment system in a traditional orientation and mobility rehabilitation program: a case study. **Disability and Rehabilitation: Assistive Technology**, Taylor & Francis, v. 7, n. 5, p. 420–435, 2012.

LOOMIS, J. M.; KLATZKY, R. L.; GIUDICE, N. A. Representing 3d space in working memory: Spatial images from vision, hearing, touch, and language. **Multisensory imagery**, New York, NY: Springer New York, p. 131–155, 2013.

LUSSAC, R. M. P. Considerações psicomotoras sobre a lateralidade e respectivos apontamentos acerca da capoeira. **CAMINHOS DA EDUCAÇÃO diálogos culturas e diversidades**, v. 4, n. 1, p. 01–13, 2022.

MACIEL, S. F. O "ir e vir" do deficiente visual (princípios, técnicas e procedimentos). **CMDV-Portal do Deficiente Visual**, 2003.

MARIA, G. D.; STACCHIO, L.; MARFIA, G. Unity-VRlines: Towards a modular extended reality unity flight simulator. In: **International Conference on Entertainment Computing**. [*S. l.*]: Singapore: Springer Nature Singapore, 2023. p. 241–250.

MARQUES, F. *et al.* Probee: a provenance-based design for an educational game analytics model. **Technology, Knowledge and Learning**, Springer, p. 1–22, 2024.

MAY, K. R.; TOMLINSON, B. J.; MA, X.; ROBERTS, P.; WALKER, B. N. Spotlights and soundscapes: On the design of mixed reality auditory environments for persons with visual impairment. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, USA, v. 13, n. 2, p. 1–47, 2020.

MENDONÇA, A. *et al.* **Alunos cegos e com baixa visão. Orientações curriculares**. [*S. l.*]: DGIDC/DSEEASE, 2008.

MORAES, J. M. *et al.* Dashboards to support rehabilitation in orientation and mobility for people who are blind. In: **Proceedings of the 29th Brazilian Symposium on Multimedia and the Web**. [*S. l.*: *s. n.*], 2023. p. 6–10.

NARASIMHAM, G. *et al.* Encoding height: Egocentric spatial memory of adults and teens in a virtual stairwell. In: **ACM Symposium on Applied Perception 2020**. [*S. l.*: *s. n.*], 2020. p. 1–8.

NOROWI, N. M.; AZMAN, H.; WAHAT, N. W. A. Apple Swipe: A Mobile game Apps for Visually Impaired Users Using Binaural Sounds. In: **Proceedings of the Asian CHI Symposium 2021**. [*S. l.*: *s. n.*], 2021. p. 196–201.

NYSTROM, R. **Game programming patterns**. [*S. l.*]: Genever Benning, 2014.

ODA, T. *et al.* VR Application for Supporting Object Recognition Considering Eye in Low Vision. In: **Proceedings of the 2020 8th International Conference on Information and Education Technology**. [*S. l.*: *s. n.*], 2020. p. 295–299.

OFTE, S. H. Right–left discrimination: Effects of handedness and educational background. **Scandinavian Journal of Psychology**, Wiley Online Library, v. 43, n. 3, p. 213–219, 2002.

ORGANIZATION, W. H. *et al.* **World report on vision**. [*S. l.*]: World Health Organization, 2019. 160 p. p.

OUMARD, C.; KREIMEIER, J.; GOETZELMANN, T. Preliminary analysis on interaction characteristics with auditive navigation in virtual environments. In: **Proceedings of the 14th PErvasive Technologies Related to Assistive Environments Conference**. [*S. l.*: *s. n.*], 2021. p. 514–520.

PARSONS, D. *et al.* A" framework" for object oriented frameworks design. In: **Proceedings Technology of Object-Oriented Languages and Systems. TOOLS 29 (Cat. No. PR00275)**. [*S. l.*]: IEEE, 1999. p. 141–151.

PARSONS, D.; RASHID, A.; TELEA, A.; SPECK, A. An architectural pattern for designing component-based application frameworks. **Software: Practice and Experience**, Wiley Online Library, v. 36, n. 2, p. 157–190, 2006.

PENUELA, R. E. G.; POREMBA, W.; TRICE, C.; AZENKOT, S. Hands-on: Using gestures to control descriptions of a virtual environment for people with visual impairments. In: **Adjunct Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology**. [*S. l.*: *s. n.*], 2022. p. 1–4.

PICCIONI, M.; FURIA, C. A.; MEYER, B. An empirical study of api usability. In: **2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement**. [*S. l.*]: IEEE, 2013. p. 5–14.

PONCIANO, T. *et al.* A generative approach for android sensor-based applications. In: **Proceedings of the Brazilian Symposium on Multimedia and the Web**. [*S. l.*: *s. n.*], 2020. p. 33–40.

REGAL, G. *et al.* Mobile location-based games to support orientation & mobility training for visually impaired students. In: **Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services**. [*S. l.*: *s. n.*], 2018. p. 1–12.

RIBEIRO, R. A. *et al.* Investigating virtual reality locomotion techniques with blind people. In: **Proceedings of the CHI Conference on Human Factors in Computing Systems**. [*S. l.*: *s. n.*], 2024. p. 1–17.

RIBEIRO, R. K. C. Orientação e mobilidade da pessoa com cegueira adquirida: os benefícios do meio aquático como facilitador da aprendizagem. **Benjamin Constant**, n. 56, 2013.

SCHMID, H. A. Systematic framework design by generalization. **Communications of the ACM**, ACM New York, NY, USA, v. 40, n. 10, p. 48–51, 1997.

SIMÕES, D.; CAVACO, S. An orientation game with 3d spatialized audio for visually impaired children. In: **Proceedings of the 11th Conference on Advances in Computer Entertainment Technology**. [*S. l.*: *s. n.*], 2014. p. 1–4.

SPEICHER, M.; HALL, B. D.; NEBELING, M. What is mixed reality? In: **Proceedings of the 2019 CHI conference on human factors in computing systems**. [*S. l.*: *s. n.*], 2019. p. 1–15.

TECHNOLOGIES, U. **Unity - Manual: Unity User Manual 2022.3 (LTS)**. 2023. Disponível em: https://docs.unity3d.com/2022.3/Documentation/Manual/. Acesso em: 9 jun. 2024.

THEVIN, L.; BRIANT, C.; BROCK, A. M. X-road: virtual reality glasses for orientation and mobility training of people with visual impairments. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, USA, v. 13, n. 2, p. 1–47, 2020.

THINUS-BLANC, C.; GAUNET, F. Representation of space in blind persons: vision as a spatial sense? **Psychological bulletin**, American Psychological Association, v. 121, n. 1, p. 20, 1997.

(W3C), W. W. W. C. **PROV-O: The PROV Ontology**. 2011. Disponível em: https://www.w3.org/TR/prov-o/. Acesso em: 27 maio 2024.

WATEMBERG, J.; CERMAK, S.; HENDERSON, A. Right-left discrimination in blind and sighted children. **Physical & Occupational Therapy in Pediatrics**, Taylor & Francis, v. 6, n. 1, p. 7–19, 1986.

WEDOFF, R. *et al.* Virtual showdown: An accessible virtual reality game with scaffolds for youth with visual impairments. In: **Proceedings of the 2019 CHI conference on human factors in computing systems**. [*S. l.*: *s. n.*], 2019. p. 1–15.

WILLIAMS, M. A. *et al.* "just let the cane hit it" how the blind and sighted see navigation differently. In: **Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility**. [*S. l.*: *s. n.*], 2014. p. 217–224.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: **Proceedings of the 18th international conference on evaluation and assessment in software engineering**. [*S. l.*: *s. n.*], 2014. p. 1–10.

WOHLIN, C. *et al.* **Experimentation in software engineering**. [*S. l.*]: Springer Science & Business Media, 2012.

YIP, B. C.; MAN, D. W. Virtual reality (VR)-based community living skills training for people with acquired brain injury: A pilot study. **Brain injury**, Taylor & Francis, v. 23, n. 13-14, p. 1017–1026, 2009.

ZHANG, H.; BABAR, M. A.; TELL, P. Identifying relevant studies in software engineering. **Information and Software Technology**, Elsevier, v. 53, n. 6, p. 625–637, 2011.

ZHAO, Y.; BENNETT, C. L.; BENKO, H.; CUTRELL, E.; HOLZ, C.; MORRIS, M. R.; SINCLAIR, M. Demonstration of enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation. In: **Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems**. [*S. l.*: *s. n.*], 2018. p. 1–4.

ZHAO, Y. *et al.* Enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation. In: **Proceedings of the 2018 CHI conference on human factors in computing systems**. [*S. l.*: *s. n.*], 2018. p. 1–14.

APPENDIX A – SYSTEMATIC MAPPING RESULTS

Systematic Mapping Results

General Data

Title	Published At	Authors	University	Ref.	Cit	Keywords
An Orientation game with 3D spatialized audio for visually impaired children	ACE '14	Diogo Simões, Sofia Cavaco	Universidade Nova de Lisboa	7	4	Educational Game, Orientation and Mobility, Android Game, Visual Impairment
Apple Swipe: A Mobile Game Apps for Visually Impaired Users Using Binaural Sounds	Asian CHI Symposium 2021	Noris Mohd Norowi, Habibunnajar Azman, Nor Wahiza Abdul Wahat	Universiti Putra Malaysia	15	1	Mobile Games, Binaural Sounds, Visually Impaired Users
Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	CHI 2018	Yuhang Zhao, Cynthia L. Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, Mike Sinclair	Cornell University, University of Washington, Microsoft Research	6	7	Virtual Reality, White Cane, Blindness, Visual Impairments, Haptic Feedback, Auditory Feedback, Mobility
Echolocation as a Means for People with Visual Impairment (PVI) to Acquire Spatial Knowledge of Virtual Space	ACM Transactions on Accessible Computing	Ronny Andrade, Jenny Waycott, Steven Baker, Frank Vetere	The University of Melbourne	98	12	Echolocation, Visual Impairment, Virtual World, Exploration
Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	CHI 2018	Yuhang Zhao, Cynthia L. Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, Mike Sinclair	Cornell University, University of Washington, Microsoft Research	67	84	Virtual Reality, White Cane, Blindness, Visual Impairments, Haptic Feedback, Auditory Feedback, Mobility
Evaluating micro-guidance sonification methods in manual tasks for Blind and Visually Impaired people	OzCHI '22	Renan Guarese, Ron van Schyndel, Fadia Zambetta	Royal Melbourne Institute of Technology	66	0	Micro-guidance, Assistive Technologies, Blind and Visually Impaired Guidance, Mixed Reality Accessibility
FeelVR: Haptic Exploration of Virtual Objects	PETRA '18	Julian Kreimeier, Timo Gotzelmann	Nuremberg Institute of Technology	19	5	Virtual Reality, Haptic, Exploration, User Study
Hands-On: Using Gestures to Control Descriptions of a Virtual Environment for People with Visual Impairments	UIST '22	Ricardo E. Gonzalez Penuela, Wren Poremba, Christina Trice, Shiri Azenkot	Cornell University, University of Maryland, Community College of Baltimore County	20	3	Virtual Reality, Visual Impairments, Accessibility, Audio Feedback, Haptic Feedback, Hand Gestures
Inclusive Social Virtual Environments: Exploring the Acceptability of Different Navigation and Awareness Techniques	CHI EA '23	Inês Gonçalves, André Rodrigues, Tiago Guerreiro, João Guerreiro	Universidade de Lisboa	22	1	Accessibility, Social Virtual Environments, Blind, Social Acceptability, Nonvisual Interaction
Mobile Location-Based Games to Support Orientation Mobility Training for Visually Impaired Students	MobileHCI '18	George Regal, Elke Mattheiss, David Sellitsch, Manfred Tscheligi	Australian Institute of Technology, University of Salzburg	32	8	Location-based Games, Serious Games, Visually Impaired, Orientation and Mobility Training
"My Zelda Cane": Strategies Used by Blind Players to Play Visual-Centric Digital Games	CHI '23	David Gonçalves, Manuel Piçarra, Pedro Pais, João Guerreiro, André Rodrigues	Universidade de Lisboa	53	5	Blind, Accessibility, Gaming, Digital Games, Navigations
Planning Your Journey in Audio: Design and Evaluation of Auditory Route Overviews	ACM Transactions on Accessible Computing	Nida Aziz, Tony Stockman, Rebecca Stewart	Queen Mary University of London, Imperial College London	59	4	Blind Navigation, User-led Design, Auditory Route Overviews, Auditory Display Design
Sonic-Badminton: Audio-Augmented Badminton Game for Blind People	CHI 2016	Shin Kim, Kun-pyo Lee, Tek-Jin Nam	KAIST	17	15	Assistive Technology, Audio-augmentation, Blinds, Game, Badminton
Spotlights and Soundscapes: On the Design of Mixed Reality Auditory Environments for Persons with Visual Impairment	ACM Transactions on Accessible Computing	Keenan R. May, Brianna J. Tomlinson, Xiaomeng Ma, Phillip Roberts, Bruce N. Walker	Georgia Institute of Technology	107	11	Navigation, Auditory Displays, Cognitive Maps
Virtual Showdown: An Accessible Virtual Reality Game with Scaffolds for Youth with Visual Impairments	CHI 2019	Ryan Wedoff, Lindsay Ball, Amelia Wang, Yi Xuan Khoo, Lauren Lieberman, Kyle Rector	Microsoft, University of Iowa, State University of New York, University of Washington	59	39	Virtual Reality, Blind, Low Vision, Youth, Spatial Audio, Haptics
VR Application for Supporting Object Recognition Considering Eye in Low Vision	ICIET 2020	Tetsuya Oda, Hiroaki Shirai, Kyohei Toyoshima, Masaharu Hirota, Ryo Ozaki, Kengo Katayama	Okayama University of Science	23	0	Low Vision, VR, Object Recognition
VRBubble: Enhancing Peripheral Awareness of Avatars for People with Visual Impairments in Social Virtual Reality	ASSETS '22	Tiger Ji, Brianna R. Cochran, Yuhang Zhao	University of Wisconsin-Madison	87	6	Visual Impairments, Social Virtual Reality, Proxemics, Audio Feedback
VStroll: An Audio-based Virtual Exploration to Encourage Walking among People with Vision Impairments	ASSETS '21	Gesu India, Mohit Jain, Pallav Karya, Nirmalendu Diwakar, Manohar Swaminathan	Microsoft Research	58	5	Accessibility, Blind, Smartphone, Exercise, Maps, Point of Interest, Spatial Audio, Navigation, Walk
X-Road: Virtual Reality Glasses for Orientation and Mobility Training of People with Visual Impairments	ACM Transactions on Accessible Computing	Lauren Thevin, Carine Briant, Anke M. Brock	Inria Bordeaux LMU Munique, IRSA, ENAC, Université Toulouse	78	20	Accessibility, Visual Impairment, Virtual Reality, Augmented Reality, Mobility Training

Systematic Mapping Results

Developed Works

Title	Artifact(s)	Domain	Platform(s)	Engine	Features
An Orientation game with 3D spatialized audio for visually impaired children	Educational game for PVI	Orientation	Android	Unity	Gyroscope, Spatial Audio
Apple Swipe: A Mobile Game Apps for Visually Impaired Users Using Binaural Sounds	Apple Swipe: Mobile game designed for PVI using Left-Right Discrimination	Rehabilitation	Android	Unity	Binaural Audio, Gesture
Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Canetroller: controller for simulating cane interactions	Navigation	HTC Vive	N/A	Spatial Audio, Tactile Feedback, Vibrotactile Feedback, Force Feedback
Echolocation as a Means for People with Visual Impairment (PVI) to Acquire Spatial Knowledge of Virtual Space	Virtual world to explore echolocation use cases	Assistive Technology	PC	Unity	Spatial Audio
Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Canetroller: controller for simulating cane interactions[1]; Virtual environments to evaluate Canetroller[2];	Navigation	HTC Vive[2]	Unity[2]	Spatial Audio, Tactile Feedback, Vibrotactile Feedback, Force Feedback
Evaluating micro-guidance sonification methods in manual tasks for Blind and Visually Impaired people	Mixed Reality environment with sonification techniques	Assistive Technology	PC	Unity	Binaural Audio, VoiceOver
FeelVR: Haptic Exploration of Virtual Objects	Virtual environment for simulation of shapes	Assistive Technology	HTC Vive	Unity	Haptic Feedback
Hands-On: Using Gestures to Control Descriptions of a Virtual Environment for People with Visual Impairments	Haptic glove based on the LucidVR open-source project	Assistive Technology	N/A	N/A	Haptic Feedback, Gestures
Inclusive Social Virtual Environments: Exploring the Acceptability of Different Navigation and Awareness Techniques	2D Virtual environments to evaluate social virtual environments	Navigation	PC	Unity	Spatial Audio, TTS
Mobile Location-Based Games to Support Orientation Mobility Training for Visually Impaired Students	Treasure Hunt based on a physical[1] ou digital[2] game;	Rehabilitation	PC[1], Android[2]	Unity[1,2]	NFC[2], TTS[1,2]
"My Zelda Cane": Strategies Used by Blind Players to Play Visual-Centric Digital Games	Observations of how PVI navigate complex virtual environments with limited accessibility features;	Navigation	N/A	N/A	N/A
Planning Your Journey in Audio: Design and Evaluation of Auditory Route Overviews	Three soundscape audio clips describing routes to explore; Software tool to automatically generate sound summaries from routes;	Navigation	Mac[2]	N/A	Binaural Audio[1,2], Procedural Generation of Routes in Maps[2], TTS[2]
Sonic-Badminton: Audio-Augmented Badminton Game for Blind People	Sonic-Badminton: Mixed Reality game based on Badminton with a virtual shuttlecock;	Rehabilitation	Arduino / PC	Processing	Binaural Audio, Motion Sensors, Game Servers
Spotlights and Soundscapes: On the Design of Mixed Reality Auditory Environments for Persons with Visual Impairment	VR environment based on one Section of the University[1]; Improved version of the same environment for Mixed Reality[2]	Assistive Technology	HTC Vive[1], iOS[2]	Unity[1,2]	Customization[2], TTS[1,2]
Virtual Showdown: An Accessible Virtual Reality Game with Scaffolds for Youth with Visual Impairments	Virtual Showdown: digital adaptation of the game Showdown, using hybrid environments with 7 stages	Rehabilitation	PC	Unity	Spatial Audio, In-game Analytics Collection, Progressive Difficulty, Haptic Feedback, Multimodal Feedback, Hint System
VR Application for Supporting Object Recognition Considering Eye in Low Vision	Shooting game applying visual correction for People with Low Vision;	Rehabilitation	Oculus Rift + Touch	Unity	Automatic Visual Correction
VRBubble: Enhancing Peripheral Awareness of Avatars for People with Visual Impairments in Social Virtual Reality	VRBubble: VR application that divides the environment in different social spaces	Assistive Technology	PC	N/A	Spatial Audio, Customization
VStroll: An Audio-based Virtual Exploration to Encourage Walking among People with Vision Impairments	VStroll: App that encourages PVI to walk	Rehabilitation	Android	N/A	Spatial Audio, TTS
X-Road: Virtual Reality Glasses for Orientation and Mobility Training of People with Visual Impairments	AR Cube Hunter[1]: Mixed Reality application to find a cube in a virtual room; X-Road[2]: VR application to perform om tasks in virtual environments	Rehabilitation	Android[1,2], PC[2], HTC Vive[2], Google Cardboard[2]	Unity[1,2]	Binaural Audio[1,2]

Systematic Mapping Results

Methodology

Title	Methodology
An Orientation game with 3D spatialized audio for visually impaired children	Development of a digital game; User playtesting sessions;
Apple Swipe: A Mobile Game Apps for Visually Impaired Users Using Binaural Sounds	Development of Apple Swipe; User playtesting sessions;
Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Development of a physical controller for PVI;
Echolocation as a Means for People with Visual Impairment (PVI) to Acquire Spatial Knowledge of Virtual Space	Lab experiment exploring echolocation in virtual environments through user testing in two stages: exploring landmarks, followed by a survey; Accessibility test of the virtual environment used (with the collection of qualitative and quantitative metrics);
Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Formative Study; Development of a physical controller for PVI; Evaluation with users of the tool's use cases in a virtual environment;
Evaluating micro-guidance sonification methods in manual tasks for Blind and Visually Impaired people	Evaluation of Hypothesis; Development of scens with sonification techniques; User testing with metrics collection;
FeelVR: Haptic Exploration of Virtual Objects	User experiment;
Hands-On: Using Gestures to Control Descriptions of a Virtual Environment for People with Visual Impairments	Development of gestures to aid navigation in a virtual environment;
Inclusive Social Virtual Environments: Exploring the Acceptability of Different Navigation and Awareness Techniques	Development of Social Virtual Environments; User testing;
Mobile Location-Based Games to Support Orientation Mobility Training for Visually Impaired Students	Game development process for a PC/Mobile game using user-centered design; Comparative study between the PC and Android versions; Interviews with om instructors;
"My Zelda Cane": Strategies Used by Blind Players to Play Visual-Centric Digital Games	Systematic Analysis of Video Footage;
Planning Your Journey in Audio: Design and Evaluation of Auditory Route Overviews	Preliminar questionnaire with PVI; Design workshop with audio experts; Development of sound routes; User testing; Development of the software tool; Usability testing;
Sonic-Badminton: Audio-Augmented Badminton Game for Blind People	Game development process for Sonic-Badminton; User playtesting sessions;
Spotlights and Soundscapes: On the Design of Mixed Reality Auditory Environments for Persons with Visual Impairment	Developing sound models and scenes for their respective use cases; Two experiments to explore the sound models in virtual environments;
Virtual Showdown: An Accessible Virtual Reality Game with Scaffolds for Youth with Visual Impairments	Game development process of Virtual Showdown; User playtesting sessions;
VR Application for Supporting Object Recognition Considering Eye in Low Vision	Development of the solution; User testing;
VRBubble: Enhancing Peripheral Awareness of Avatars for People with Visual Impairments in Social Virtual Reality	Development of the technique for virtual environments with user-centered design;
VStroll: An Audio-based Virtual Exploration to Encourage Walking among People with Vision Impairments	Development process of VStroll; User testing;
X-Road: Virtual Reality Glasses for Orientation and Mobility Training of People with Visual Impairments	Development process of X-Road using participative design; User testing;

Systematic Mapping Results

Experiment

Title	Experiment(s)	Participantes	PVI	Procedure
An Orientation game with 3D spatialized audio for visually impaired children	Preliminary test of the first version of the game	5	5	Prior Use Instructions, Game Session (with direct help in the first stages and when having users are having difficulties to use)
Apple Swipe: A Mobile Game Apps for Visually Impaired Users Using Binaural Sounds	Game Playtest of Apple Swipe with PVI[1]; Applying the same playtest with people without visual impairments[2];	2[1] 6[2]	2tb[1] 0[2]	Game Session
Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	N/A	N/A	N/A	N/A
Echolocation as a Means for People with Visual Impairment (PVI) to Acquire Spatial Knowledge of Virtual Space	Performing echolocation tasks to test it's use[1]; Reconstruction task with a virtual environment[2];	12	12 (9tb, 2blp, 1lv)	Prior Instructions of Use, Completing Missions with "yes-no" assistances
Semistructured Interview(s)[1];				
Exploring the Virtual Environment				
Reconstruction Task	Semistructured Interview(s)[2]."			
Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Evaluation of Canetroller by cane users	9	9	Interview, Prior Exploration of the Physical Space, Prior Instructions for Use, Exploration of an Indoor Environment, Exploration of an Outdoor Environment, Interview
Evaluating micro-guidance sonification methods in manual tasks for Blind and Visually Impaired people	Evaluation of all sonification techniques presented	47	9 (5tb, 4 lv)	Prior Instructions of Use, Performing Goals
FeelVR: Haptic Exploration of Virtual Objects	Identifying virtual shapes using a physical controller (with and without the use of vision)	10	0	Prior Instructions of Use, Prior Exploration of Controls, Completing tasks, Completing the same tasks again, but without vision, Conversation with Participant
Hands-On: Using Gestures to Control Descriptions of a Virtual Environment for People with Visual Impairments	N/A	N/A	N/A	N/A
Inclusive Social Virtual Environments: Exploring the Acceptability of Different Navigation and Awareness Techniques	Testing multiple navigation and interaction methods in virtual environment.	16	8	Initial Presentation, Consent Form, Demographic Questionnaire, Exploration of the Environment (hints available to the user at any moment), Questionnaire, Semistructured Interview(s)
Mobile Location-Based Games to Support Orientation Mobility Training for Visually Impaired Students	Evaluation of the digital game for PC[1] (virtual environment); Evaluation of the mobile game for Mixed Reality[2]; Interviews with om instructors[3];	12[1] 15[2] 3[3]	12[1] 15[2] 0[3]	Initial Presentation, Gameplay Session (with assistance)[1,2], Questionnaire[1,2]; Introduction to the Game Editor (with example), Semistructured Interview(s)[3]
"My Zelda Cane": Strategies Used by Blind Players to Play Visual-Centric Digital Games	N/A	N/A	N/A	N/A
Planning Your Journey in Audio: Design and Evaluation of Auditory Route Overviews	Preliminary analysis with route overviews[1]; Evaluation the descriptions of prebaked sound clips for routes[2]; Study of the effectiveness of the sound route generating system[3];	15[1] 8[2] 28[3]	15[1] 8[2] 6[3]	Questionnaire[1]; Listening session, Questionnaire[2]; Prior Instructions of Use, Reconstruction Task, Recognition Task, Questionnaire[3];
Sonic-Badminton: Audio-Augmented Badminton Game for Blind People	Evaluation of Sonic-Badminton	6	3	Prior Interview, Table Tennis session, Gameplay Session of Sonic-Badminton, Group Interview
Spotlights and Soundscapes: On the Design of Mixed Reality Auditory Environments for Persons with Visual Impairment	Exploring a soundscape in a VR environment to raise requirements[1] Evaluation of an Mixed Reality environment created using the first experiment[2]	5[1] 5[2]	5[1] 5[2]	Consent Form, Prior Instructions of Use, Performing Tasks, Exploring the virtual world (active objects), Evaluation Scales (SUS, BUZZ), Exploring the environment (isolated objects), Semistructured Interview(s) (during the last stop), Reapplying the evaluation scales (SUS, BUZZ), Demographic Questionnaire[1]; Consent Form, Navigation Tasks, NASA-TLX Questionnaire, Reconstruction Tasks, BUZZ Evaluation Scale, Interview, Free Exploration of the solution[2]
Virtual Showdown: An Accessible Virtual Reality Game with Scaffolds for Youth with Visual Impairments	Evaluation of Virtual Showdown game	34	29	Initial Presentation, Game Session (6 minutes), Interview
VR Application for Supporting Object Recognition Considering Eye in Low Vision	Evaluation of visual correction through a demonstration	1	1	Gameplay Session (1 minute, 10 session)
VRBubble: Enhancing Peripheral Awareness of Avatars for People with Visual Impairments in Social Virtual Reality	Evaluating VRBubble with virtual environments for demonstration purposes	12	12	Interview, Prior Instructions of Use, Navigation Tasks (questions at the end), Exploration Tasks (questions asked during session), Conversation with Participant
VStroll: An Audio-based Virtual Exploration to Encourage Walking among People with Vision Impairments	Evaluating VStroll with users for 5 days;	16	16tb	Consent Form, Prior Instructions of Use, Data Collection (through in-game analytics), Semistructured Interview(s) after use
X-Road: Virtual Reality Glasses for Orientation and Mobility Training of People with Visual Impairments	Evaluation of X-Road	13	13 (4tb, 2pc, 7lv)	Consent Form, Initial Presentation, Questionnaire, Session with AR Cube Hunter, Tasks performed with X-Road, Questionnaire, Exploration of X-Road using Google Cardboard

Systematic Mapping Results

Results and Limitations

Title	Results	Limitations
An Orientation game with 3D spatialized audio for visually impaired children	Game was approved by participants; Immediate positive impact on motor coordination; Initial learning curve related to the body's rotation; Audio panning is not enough to establish spatialization, spatial audio is recommended;	N/A
Apple Swipe: A Mobile Game Apps for Visually Impaired Users Using Binaural Sounds	A Game was approved by participants; PVI had the highest hit rate; Device's orientation, interactions and graphic interface were different from what participants expected; Lack of TTS (recommanded by players);	N/A
Demonstration of Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Use cane skills in the virtual world;	N/A
Echolocation as a Means for People with Visual Impairment (PVI) to Acquire Spatial Knowledge of Virtual Space	Higher sucess rate when identifying room sizes, materials and 90 degree turns using life experiences; Level progression, with the first three stages being easier in comparison; No correlations between artificial pulses and participants' success rate; Echo generated by steps was preferred over artificial pulses; Multiple techniques for describing the virtual world (perimeter recognition, listing elements, describe holistic map models); Sense of presence in the virtual environment; Over time, skills improve when interacting in the virtual world; Participants were interested in virtual worlds for entertainment purposes;	N/A
Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation	Faster and more efficient use of skills in the virtual world; Instructors were positive about the use of Canetroller in om training; Force Feedback is limited to the horizontal axis; Difficulty perceiving changes in surface altitude; Limited Physical space for exploring big virtual worlds; Difficulties in use due to the equipment's weight;	N/A
Evaluating micro-guidance sonification methods in manual tasks for Blind and Visually Impaired people	Spatialized Pitch is more reliable than other Pitch techniques; The most efficient approach is combining spatialization with alternance; Harmonics highly depends on the user's knowledge of music theory; Vision does not affect accuracy in microorientation tasks, but can affect runtime; Avoid using methods that depend on auditive skills (e.g. musical literacy); User preferences for listening audio at faster speeds impacted their performance in distance tasks;	N/A
FeelVR: Haptic Exploration of Virtual Objects	More distinct shapes (cone, cube) are more easily identified; Lack of vision helped with the participants' accuracy; The stance used for controls limited the shape recognition process;	N/A
Hands-On: Using Gestures to Control Descriptions of a Virtual Environment for People with Visual Impairments	Set of commands to help PVI explore virtual environments;	N/A
Inclusive Social Virtual Environments: Exploring the Acceptability of Different Navigation and Awareness Techniques	Autonomy is preferred over the method's efficiency; There's no favorite methods, but preference for flexible navigation: Teleport is the most efficient, while Free Exploration is both the least efficient and the most accepted; Audio feedback is a must for PVI, redundant for people without visual impairments; Participants value having access to information, even when the elements are far away;	N/A
Mobile Location-Based Games to Support Orientation Mobility Training for Visually Impaired Students	Playing in the real world is more fun than playing in the virtual one; Implementation may have biased the comparitive study's results; Game is great for practicing om skills, but is not a replacement for basic om training, requiring prior knowledge of it's fundamentals; Description Feature was the most well received, but current technology used for this work does not scale well to more complex environments;	N/A
"My Zelda Cane": Strategies Used by Blind Players to Play Visual-Centric Digital Games	Use of landmarks (natural or user-defined) to represent locations and build a mental map; Extra mental effort to play, similar to exploring a new location in the real world; Camera control is an obstacle for perception; A top-down perspective leads to better mental models; Use of spatialized audio and soundscapes facilitate the distinction between elements; Design decisions from a game offer a limited set of unique sounds; Learning patterns tend to be most exclusively visual; Use of the game's features as tools for interaction and navigation; Need for mechanisms that provide information in parallel (e.g. through sound and haptic channels); Use of cooperative strategies such as sharing a controller in order to give PVI more agency; Accessibility gaps can be mitigated by mechanisms to allow game modification and expansion; Searching for the best path is the same as searching for the best skill loadout for a character; Make older titles more accessible through mods and guides;	Results obtained are not a representation of the general PVI population; Some navigation approaches were not encountered (e.g. side-scrolling games); The research did not evaluate haptic feedback in the process;
Planning Your Journey in Audio: Design and Evaluation of Auditory Route Overviews	Sound descriptions are useful for PVI to navigate independently; Participants preferred route summaries over other summary types; Summary duration and techniques used vary based on the scope of the content being described; Main and Functional information transmitted through speech, Audio Icons and earcons as environmental information; Interest in customizing elements and level of detail for information; Some Audio Icons confused participants; Sound intervals to describe a location are well represented, but are hard to place in the scene; Over time, participants' performance improve when using the solution;	App only allows passive interactions with routes; Usability test was limited to allow for a laboratory study; Usability test contains unbalanced samples between the groups;
Sonic-Badminton: Audio-Augmented Badminton Game for Blind People	PVI highlight the combined use of audio feedback with the physical activity; Opens new possibilities for PVI to play previously unaccessible games; Allow for an experience where PVI and people without visual impairments play at the same level.	Lack of tactile feedback in the final solution;
Spotlights and Soundscapes: On the Design of Mixed Reality Auditory Environments for Persons with Visual Impairment	The Spotlights and Soundscapes presented were approved by PVI participants in terms of usability, optimizations, learning curve and creating cognitive maps; Spotlights are ideal to filter information, add predictability to directions; Soundscapes are ideal to give ample context about the environment; Models used were too heavy on information to be processed by PVI Object representations with implicit information makes it easier to build a mental map; Need to curate objects and descriptions in environments to determine which are the most relevant to the user based on the context; Spatial audio can't be the only way to transmit an information; Sound beacons don't make clear when and how to change direction; Realistic sound in the virtual world can be interpreted as sounds in the real world; Need for system customization elements for use;	N/A
Virtual Showdown: An Accessible Virtual Reality Game with Scaffolds for Youth with Visual Impairments	Previous experience with Showdown affected the final scores proportionally; PVI scored better when the ball stayed on the same side as the dominant hand when compared to a diagonal trajectory and when the ball returned in the center of the table versus returning on the side opposite to the dominant hand; The lack of a height requirement for the hand's position improves accessibility; Showdown players prefer verbal cues instead of having verbal and haptic cues in parallel; Some motivations from participants to play again include: challenge, experience, an even playing field with people without visual impairments, novelty factor, physical activity and sharing the experience with friends; Feedback used to represent hitting the ball back was unclear;	N/A
VR Application for Supporting Object Recognition Considering Eye in Low Vision	The visual correction applied improved the player's accuracy;	N/A
VRBubble: Enhancing Peripheral Awareness of Avatars for People with Visual Impairments in Social Virtual Reality	Improvement of the peripheral perception of avatars; Earcons have an initial learning curve; Representing avatar information requires more intuitive sounds; Verbal descriptions are clear and easy to understand, but during a conversation it may cause distractions; Participants were selected for the sound feedback based on the social context; Peripheral perception has a maximum capacity; PVI need a way to control and customize audio from the environment; Experiment acts as a use case of the technique in a virtual social environment;	N/A
VStroll: An Audio-based Virtual Exploration to Encourage Walking among People with Vision Impairments	Increase in motivation for PVI to walk, be more responsible with their lives or remember old memories; Participants added gamification elements, even with the application not having any game elements; Participants used the solution during and after the research experiment, experiencing immersion in the virtual environment; Implement features that give autonomy for PVI (e.g. voice commands); Allows PVI to acquire spatial knowledge about the explored location and motivates them to explore more unique or inacessible locations due to socioeconomic constraints; Users recommended adding new exercise monitoring features, as well as frequent and informative Points of Interest;	Low user sample size; App assumes that all roads between two intersections are straight roads; Results can't be generalized to people with low vision;
X-Road: Virtual Reality Glasses for Orientation and Mobility Training of People with Visual Impairments	Basic levels of audiovisual fidelity are enough to create accessible VR experiences for PVI; Provide multiple feedback types for similar objects; X-Road allows PVI to explore environments in a way that's not possible in the real world; AR Cube Hunter was useful in explaining generic audiospatial concepts; Extremely high levels of realism can remove the advantages brough by the virtual world due to it being confused with the real world; There's no way to distinguish between sounds in front or behind the user in VR; It's recommended to have another person when PVI are exploring virtual environments; The use of simple movement systems allows PVI to escape the expected route and look for elements in the virtual world;	

APPENDIX B – FORM GUIDE (IN PORTUGUESE)

Participação na Pesquisa “Incrementando Habilidades de Orientação e Mobilidade para Pessoas com Deficiência Visual: Introdução à biblioteca Unity EdKit.”

Eu, José Martônio Lopes de Moraes Júnior, integrante do Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat) da Universidade Federal do Ceará, gostaria de convidá-lo(a) para participar da pesquisa “Incrementando Habilidades de Orientação e Mobilidade para Pessoas com Deficiência Visual: Introdução à biblioteca Unity EdKit.” que objetiva avaliar uma biblioteca de software (EdKit), com proposta temática de estudo sobre o desenvolvimento de jogos de Realidade Virtual Imersiva para estímulo às práticas de atividades de Distinção Direita-Esquerda para usuários com deficiência visual.

O(a) senhor(a) irá acessar o projeto a ser analisado, assim que concordar com os critérios a seguir. Sua participação é importante, porém você não deve participar contra a sua vontade. Leia atentamente as informações abaixo e faça qualquer pergunta que desejar, para que todos os procedimentos desta pesquisa sejam esclarecidos.

- Os benefícios deste estudo experimental estão principalmente em investigar e especificar elementos do desenvolvimento de jogos de realidade virtual imersiva para pessoas com deficiência visual usando a biblioteca EdKit e, assim, propiciar o desenvolvimento de novos jogos para tal público trabalhando habilidades de Distinção Direita-Esquerda.

- Destaco que a qualquer momento o(a) senhor(a) poderá se recusar a continuar participando da pesquisa, sem nenhum prejuízo e mesmo em caso de aceite do referido termo. Para isso, basta que informe seu desejo de parar a atividade em andamento ou deixar o local onde será realizada a avaliação.

- Posso assegurar a privacidade no momento em questão e, para sua maior segurança, será mantido sigilo em relação ao seu nome e/ou quaisquer outros aspectos que possam vir a identificá-lo(a). As informações utilizadas neste estudo possuirão a única finalidade de colaborar com a presente pesquisa bem como a divulgação em relatórios e revistas científicas.

- É oportuno esclarecer que para participar desta pesquisa não será oferecido nenhum valor em pecúnia ou compensações similares ao(a) senhor(a). Portanto, nesta pesquisa, sua participação é totalmente voluntária.

O tempo estimado total de participação é de uma sessão única (65 minutos presencialmente, 2 horas remoto), composto por 4 atividades específicas:

- Atividade 1. Avaliar a experiência do participante com Programação para Jogos: A tarefa visa estabelecer o perfil de programador do participante para entender suas

experiências prévias com desenvolvimento de jogos, engines de jogos e realidade virtual.

- Atividade 2. Introdução à biblioteca EdKit e ao jogo de demonstração `King`: A tarefa visa apresentar: o que é a biblioteca EdKit, como a biblioteca pode ser usada no processo de desenvolvimento de jogos, introdução ao jogo de demonstração `King`, suas mecânicas e como as funcionalidades da biblioteca são aplicadas, tirando dúvidas prévias que o participante possa ter sobre as funcionalidades existentes.
- Atividade 3. Implementação de uma nova funcionalidade para `King`: A tarefa consiste na implementação de uma nova pose para o jogo seguindo uma especificação de requisitos previamente preparada. Para fazer um uso abrangente das funcionalidades do pacote, os requisitos incentivam o uso de todos os módulos da biblioteca.
- Atividade 4. Avaliar a experiência do desenvolvedor com EdKit: A tarefa envolve o preenchimento de um questionário avaliando a experiência do desenvolvedor com EdKit e a usabilidade da API.

Caso o participante tenha quaisquer dúvidas durante o processo, o mesmo poderá requisitar auxílio do condutor da pesquisa a qualquer momento, estando o mesmo à disposição para sanar as dúvidas apresentadas.

Espero poder contar com sua inestimável cooperação e desde já agradeço pela atenção.

* Indicates required question

1. Nome *

Consentimento

Ao avançar neste questionário, declaro que possuo idade maior que 18 anos e compreendi perfeitamente tudo que me foi informado sobre minha participação no mencionado estudo e, estando consciente dos meus direitos, das minhas responsabilidades, dos riscos e dos benefícios que a minha participação implica, concordo em dele participar e para isso dou o meu consentimento sem que tenha sido obrigado.

Experiência com Programação de Jogos

2. Quanto tempo de experiência você possui com Programação para Jogos? *

Mark only one oval.

- ◯ <1 ano
- ◯ 1-2 anos
- ◯ 3-5 anos
- ◯ 5+ anos

3. Quanto tempo de experiência você possui com as seguintes engines de jogos? *

Mark only one oval per row.

	<1 ano	1-2 anos	3-5 anos	5+ anos	Desconheço / Não possuo experiência
Unity	◯	◯	◯	◯	◯
Unreal Engine	◯	◯	◯	◯	◯
Godot Engine	◯	◯	◯	◯	◯
Engine Própria	◯	◯	◯	◯	◯
Outras	◯	◯	◯	◯	◯

4. Quanto tempo de experiência você possui com as seguintes linguagens de programação? *

Mark only one oval per row.

	<1 ano	1-2 anos	3-5 anos	5+ anos	Desconheço / Não possuo experiência
C#	◯	◯	◯	◯	◯
C++	◯	◯	◯	◯	◯
GDScript	◯	◯	◯	◯	◯
C	◯	◯	◯	◯	◯
Outras	◯	◯	◯	◯	◯

5. Quanto tempo de experiência você possui com desenvolvimento para Realidade Virtual?

   Mark only one oval.

   - ◯ <1 ano
   - ◯ 1-2 anos
   - ◯ 3-5 anos
   - ◯ 5+ anos
   - ◯ Não possuo experiência

Construindo um Jogo usando EdKit

Agora o condutor da pesquisa irá lhe acompanhar na realização das atividades de Introdução à biblioteca EdKit e ao jogo de demonstração `King`, assim como a Implementação de uma nova funcionalidade para `King` seguindo o guia disponível abaixo.

Link: https://martjun10r.notion.site/Construindo-um-Jogo-usando-EdKit-8a71a9f4307340f88f0450ab88233e98

Ao terminar, responda as questões a seguir para avançar à próxima sessão.

6. Constato que, na minha perspectiva... *

   Check all that apply.

   - ☐ Implementei um novo comando chamado "Bloqueio"
   - ☐ Adicionei o comando "Bloqueio" a lista de comandos identificáveis no jogo
   - ☐ Criei uma nova Affordance para representar o comando "Bloqueio"
   - ☐ Conectei o comando "Bloqueio" e sua Affordance ao sistema de requisição de comandos
   - ☐ Criei uma nova atividade "Block" para o comando "Bloqueio"
   - ☐ Registrei a nova atividade "Block" na sessão de jogo
   - ☐ Não consegui realizar nenhuma das atividades propostas

7. Entendi claramente como funciona... *

Mark only one oval per row.

	1 - Discordo Totalmente	2 - Discordo	3 - Neutro	4 - Concordo	5 - Concordo Totalmente
EdKit	◯	◯	◯	◯	◯
Jogo `King`	◯	◯	◯	◯	◯
Nova Feature	◯	◯	◯	◯	◯

8. Requisitei a ajuda do condutor da pesquisa para... *

Check all that apply.

- [ ] Implementar um novo comando chamado "Bloqueio"
- [ ] Adicionar o comando "Bloqueio" a lista de comandos identificáveis no jogo
- [ ] Criar uma nova Affordance para representar o comando "Bloqueio"
- [ ] Conectar o comando "Bloqueio" e sua Affordance ao sistema de requisição de comandos
- [ ] Criar uma nova atividade "Block" para o comando "Bloqueio"
- [ ] Registrar a nova atividade "Block" na sessão de jogo
- [ ] Esclarecer detalhes sobre os requisitos apresentados para a tarefa
- [ ] Tirar dúvidas sobre o funcionamento da biblioteca `EdKit`
- [ ] Tirar dúvidas sobre o funcionamento do jogo de demonstração `King`
- [ ] Tirar dúvidas sobre como a Unity funciona, seja no Editor ou na implementação de código
- [ ] Não pedi ajuda para o condutor

Pós-Experimento

9. A sintaxe do EdKit ajudou você a desenvolver jogos em Unity? *

Mark only one oval.

	1	2	3	4	5	
Disc	◯	◯	◯	◯	◯	Concordo Totalmente

10. Tive que desistir de implementar parte da funcionalidade pois não sabia se conseguiria implementar usando EdKit *

Mark only one oval.

	1	2	3	4	5	
Disc	◯	◯	◯	◯	◯	Concordo Totalmente

11. Responda o seu grau de concordância em relação às afirmações abaixo: *

Mark only one oval per row.

	1 - Discordo completamente	2 - Discordo	3 - Neutro	4 - Concordo	5 - Concordo completamente
EdKit torna o design mais fácil?					
EdKit torna a implementação mais fácil?					
EdKit é difícil de usar?					
EdKit é poderoso?					
EdKit não tem todas as características que eu preciso?					
EdKit restringe minha liberdade como programador?					
O código é mais legível com EdKit?					
Menos erros ocorrem com EdKit?					
EdKit cumpre melhor com as especificações pedidas?					

12. Quais são as vantagens que você viu no uso do EdKit? *

13. Quais são as desvantagens que você viu no uso do EdKit? *

14. No geral, como você avalia sua experiência com a biblioteca EdKit? *

Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Muit	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muito Bom

15. Você tem alguma sugestão ou comentário sobre o EdKit?

Construindo um Jogo usando EdKit

> *Neste tutorial, você irá aprender sobre uma das amostras de EdKit e como o pacote pode ser usado no processo de desenvolvimento de jogos de Realidade Virtual para Pessoas com Deficiência Visual.*

Requer Unity 2022 LTS ou superior (Recomendação: Unity 2022.3.48f1)

https://drive.google.com/file/d/1-t2cSpGR0nVVW4wpRmqekfgU_o_hPLU2/view?usp=sharing

Dica: Baixe o projeto e abra-o ao menos uma vez para preparar o ambiente do experimento.

King

> *Jogo para um jogador em desenvolvimento para Meta Quest 2, onde pessoas com deficiência visual podem testar suas habilidades de distinção de planos, reconhecimento de pistas, comandos alinhados e coordenação motora.*
> *Nele, o jogador precisa obedecer aos comandos anunciados para ganhar pontos e sobreviver o máximo de tempo possível.*

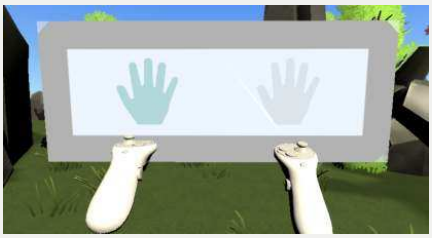


Figura 1 - Jogabilidade de `King`

Objetivo: Conseguir a maior pontuação possível.

Controles

- Levantar a Mão Esquerda/Direita acima da cabeça para levantar uma bandeira
- Start: Pausa o jogo
- Grip Direito: Reinicia o jogo atual

Como Jogar?

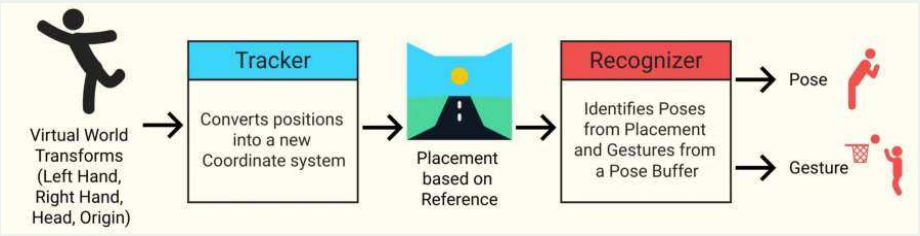
Ao iniciar, o jogo pede para que a bandeira em uma das mãos seja levantada pelo jogador:

- Ao acertar, o jogador ganha 1 ponto e o jogo anuncia um novo comando a ser feito
- Ao errar, o jogador perde uma vida e o jogo pede para tentar novamente o mesmo comando

O jogo começa com o jogador tendo acesso à três vidas, e termina quando todas as vidas forem perdidas.

Sobre EdKit

Gestures



`Axis`: *Estrutura de dados descrevendo a diferença espacial de alinhamento entre dois objetos (ex: vetor distância $\overrightarrow{ab}$)*

Axis	Antes	Centro
`Axis.X`	`Left`	`Center`
`Axis.Y`	`Low`	`Neutra`
`Axis.Z`	`Back`	`Body`

- Inicialização pode incluir uma "deadzone" para definir a área que representa o Centro

```
classDiagram
direction TB

class X["Axis.X"] {
    <<enumeration>>
    None
    Left
    Center
    Right
}

class Y["Axis.Y"] {
    <<enumeration>>
    None
    Below
    Neutral
    Above
}

class Z["Axis.Z"] {
    <<enumeration>>
    None
    Behind
    Body
    Front
}

X <.. Axis
Y <.. Axis
```

`Orientation`: *Estrutura de dados contendo a posição e rotação relativas de um objeto*

- Usa `Bounds` para obter valores normalizados baseado no centro (0) e nas bordas (-1/+1)
- Pode ser comparado com outras instâncias de `Orientation`, retornando um `Axis`.
- Pode posicionar Transforms em relação a si mesmo para criar novas instâncias de `Orientation` e `Placement`

`Placement`: *Representação do posicionamento do jogador em VR, composto por cabeça e mãos esquerda e direita*

- Criado a partir da interpretação de 4 referências de `Transform`: Cabeça, Mão Esquerda, Mão Direita e Origem
- Pode ser utilizado por qualquer ator de jogo com elementos equivalentes (cabeça → corpo, mão → pé, etc.)

```
Z <.. Axis
class Axis {
    <<struct>>
    +var x: Axis.X
    +var y: Axis.Y
    +var z: Axis.Z

    +getX(delta: Vector3, dea
    +getY(delta: Vector3, dea
    +getZ(delta: Vector3, dea
}

Axis <.. Orientation
class Orientation {
    <<struct>>
    +var position: Vector3
    +var rotation: Vector3

    +comparePosition(to orier
    +compareRotation(to orier
    +normalizedPosition(in bc
    +normalizedRotation(in bc
    +place(_ transform: Trans
    +placement(leftHead: Trar
}

Orientation <.. Placement
class Placement {
    <<struct>>
    +var leftHand: Orientatic
    +var rightHand: Orientati
    +var head: Orientation
}
```

```
classDiagram
direction TB

class IPose {
    <<interface>>
    +var name: String*
```

`Event` : *Estrutura de dados descrevendo uma avaliação de similaridade em EdKit*

- Objeto: Elemento associado ao evento
- Dados: Informação sobre o evento
- Score: Pontuação dada a similaridade entre objeto e dados

Score ≤ 0	0 < Score < 1	Score ≥
Evento não será criado	Evento é criado com o Score especificado: quanto mais perto de 1, maior é a similaridade.	Evento com Sc objeto (conside equival dados.

- Contém mecanismos de filtro usando limiares para a pontuação

```
    +evaluate(placement: Plac
}

IPose <|.. Pose
class Pose {
    <<struct>>
    -var name: String
    -var scoreFunction: Weigh

    +Pose(_ name: String, scc

    +evaluate(_ placement: Pl
}

class IGesture {
    <<interface>>
    +var name: String*

    +evaluate(_ poses: PoseEv
}

IGesture <|.. Gesture
class Gesture {
    <<struct>>
    alias Parser = Weighted~[

    -var scoreFunction: Parse

    +Move(_ parser: Parser)
    +Move(checks: Weighted~Pc
}
```

```
classDiagram

class Tracker {
    <<component>>
    -var headTransform: Trans
    -var leftHandTransform: T
    -var rightHandTransform:
```

`Pose`: *Interpretação de um* `Placement` *com base em um algoritmo de pontuação*

- "Traga seu próprio algoritmo": o programador define o algoritmo de pontuação para um `Placement`
- Poses reconhecidas por EdKit são armazenadas como “Pose Events”

Como criar uma nova Pose em EdKit?

- Abordagem 1: Implementar usando `Pose`
  1. Criar uma nova instância de `Pose` com o algoritmo desejado
- Abordagem 2: Implementar usando `IPose`
  1. Criar um novo tipo que implemente a interface `IPose`
  2. No método `Evaluate`, implementar o algoritmo desejado

```
    -var xrTransform: Transfo
    -var offsetCenter: Vector

    +var onUpdate: Action~Pla

    +getPlacement() Placement
    +sample() Void
}

class Recognizer {
    <<component>>
    -var poseBuffer: Buffer~P

    +var onGestureIdentified:
    +var onPoseIdentified: Ev

    +register(_ placement: Pl
    +sample() GestureEvent?
    +sample(_ buffer: Buffer-
    +squash() Void
}
```

`Gesture`: *Interpretação de uma sequência de Pose Events com base em um algoritmo de pontuação*

- Assim como em `Pose`, o programador define o algoritmo de pontuação para a sequência de poses
- Gestos reconhecidos por EdKit são armazenadas como “Gesture Events”

Como criar um novo Gesto em EdKit?

- Abordagem 1: Implementar usando `Gesture`
  1. Criar uma nova instância de `Gesture` com o algoritmo desejado
- Abordagem 2: Implementar usando `IGesture`
  1. Criar um novo tipo que implemente a interface `IGesture`
  2. No método `Evaluate`, implementar o algoritmo desejado

`Tracker`: *Componente responsável por criar instâncias de* `Placement` *a partir de instâncias de* `Transform` *em uma cena*

- Atualizações acontecem em `FixedUpdate`, mas o método `Sample` pode ser usado para uma amostragem controlada

Como funciona o processo de conversão?

1. Definem-se eixos de referência, compostos pelo vetor frontal da cabeça projetado no plano XZ e o vetor para cima da origem.
2. Construção de um novo sistema de coordenadas usando os eixos de referência
3. Conversão das posições do Espaço global para o novo sistema

`Recognizer` : *Componente responsável por identificar poses e gestures usando instâncias de* `Placement`

- Contém dois mecanismos de avaliação: um para poses e outro para gestos
- Armazena um buffer contendo os últimos “Pose Events” recebidos

Como funciona o reconhecimento de poses e gestos?

1. Quando um novo `Placement` é registrado, o componente avalia o mesmo usando um conjunto de possíveis poses e seleciona a pose que resulte na maior pontuação
2. `Pose` é adicionada ao buffer como um “Pose Event” e o evento de poses é disparado
3. Após a adição de um novo “Pose Event”, o componente faz o mesmo processo de avaliação com todos os possíveis gestos e selecionar aquele com a maior pontuação
4. Cria-se um novo "Gesture Event” com o gesto e o evento de gestos é disparado
5. O buffer de "Pose Events” é esvaziado por completo

Feedback

Tipos de Feedback

- **Audio:** Representação de um feedback sonoro (ex: `AudioClip`, etc.)
- **Visual:** Representação de um feedback visual (`Color`, `Material`, `Animation`, etc.)
- **Text:** Representação de um feedback textual e/ou verbal (`String`)
- **Action:** Código executado para aplicar outros tipos de feedback (ex: háptico) ou executar outras ações relevantes

`Affordance`: *Estrutura de dados representando uma retorno de informação ao jogador*

- Mecanismo de feedback multimodal aplicado simultaneamente por canais de representação distintos

Creating a new Affordance

1. In the Project window, right-click and go to Create → EdKit → Affordance
2. Configure the `AudioClip`, `Material`, `String` and a custom `UnityEvent` to trigger when the affordance is updated

```
classDiagram
direction TB

class AffordanceEffect {
    <<struct>>
    +var alignment: Double
    +var scale: Double
    +var duration: Double

    +AffordanceEffect(_ aligr
}

AffordanceEffect <|.. IAfforc
class IAffordance {
    <<interface>>
    alias Sound
    alias Text
    alias Visual

    +var description: Text*
    +var sound: Sound*
    +var visual: Visual*

    +update(_ effect: Afforda
}
```

`AffordanceEffect`: *Estrutura de dados que fornece informação espacial à Affordance executada*

- `Duration`: Duração do efeito em execução na Affordance.
- `Alignment`: Sinal normalizado detalhando o alinhamento do jogador em um eixo.
- `Scale`: Escala para indicar o impacto do efeito em relação ao jogador.

`AffordanceReceiver`: *Componente responsável por aplicar uma Affordance na cena*

- Ao receber uma Affordance, cada representação é redirecionada para um evento diferente no componente
- Permite configurar um `AffordanceEffect` padrão no próprio componente

```
IAffordance <|.. AffordanceDa
class AffordanceData {
    <<ScriptableObject>>
    -var clip: AudioClip
    -var description: String
    -var material: Material
    -var customAction: UnityE
}

IAffordance <.. Receiver
class Receiver {
    <<component>>
    -UnityEvent~AudioClip~ or
    -UnityEvent~Material~ onV
    -UnityEvent~string~ onTex
    -UnityEvent~AffordanceEff

    +apply(_ affordance: Affo
    +update() Void
}
```

Analytics

Inspirado na ontologia PROV-O e composta das seguintes representações de proveniência:

- **Entidade (`Entity`):** Representa elementos com uma identidade persistente (ex: documentos, arquivos, dados)
- **Atividade (`Activity`):** Representa processos e ações realizadas nas entidades. Pode estar associada a `Agent` e/ou `Entity`
- **Agente (`Agent`):** Representa pessoas, organizações ou sistemas responsáveis por realizar atividades.

```
classDiagram

class Activity {
    <<struct>>
    +var activityID: String
    +var type: String
    +var attributes: [String:A
ny]
    +var used: String
    +var wasAssociatedWith: St
ring

    +init(activityID: String,
type: String, attributes: IDic
tionary~String_Any~, used: Str
ing, wasAssociatedWith: Strin
g)

    +associatedWith(_ agent: A
gent) Activity
    +usedBy(_ entity: Entity)
Activity
}

class Agent {
    <<struct>>
    +var agentID: String
    +var type: String
    +var attributes: [String:A
ny]
}

class Entity {
    <<struct>>
    +var entityID: String
    +var type: String
    +var attributes: [String:A
```

Modelo de Proveniência: *Modelo que armazena representações de proveniência*

- Informações são serializadas em um JSON
- Utiliza o padrão de design Visitor para registro dos dados de proveniência

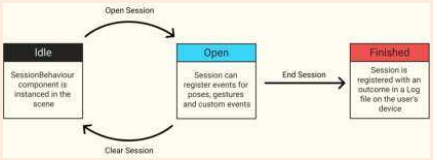
```
ny]
}

Activity <.. IProvenanceModel
Agent <.. IProvenanceModel
Entity <.. IProvenanceModel
class IProvenanceModel {
    <<interface>>
    +register(activity: Activi
ty)* Void
    +register(agent: Agent)* V
oid
    +register(entity: Entity)*
Void
    +registerOutcome(_ outcom
e: IDictionary~String_Any~, da
ta: IDictionary~String_Any~) V
oid
}

IProvenanceModel <.. IProvenan
ceData
class IProvenanceData {
    <<interface>>
    +var id: String*
    +var type: String*
    +var attributes: [String:A
ny]*

    +asActivity(used: String,
wasAssociatedWith: String) Act
ivity
    +asAgent() Agent
    +asEntity() Entity
    +register(to model: IProve
nanceModel)* Void
}

IProvenanceModel <|.. Session
class Session {
    <<struct>>
```

Dados de Proveniência:
Nome dado à estruturas de dados de proveniência

- Implementado a partir da interface `IProvenanceData`
- Estrutura controla como seus dados são registrados no modelo de proveniência

```
    +let version: UInt

    -var date: Date
    -var userID: String
    -var sceneID: String
    -var activities: [Activit
y]
    -var agents: [Agent]
    -var entities: [Entity]

    +var tag: String%

    +init(_ userID: String, _
sceneID: String, date: Date?)
}

IProvenanceModel <|.. SessionB
ehaviour
class SessionBehaviour {
    <<Component>>
    +var session: Session?

    +closeSession(outcome: Str
ing) Void
    +closeSession(outcome: IDi
ctionary~String_Any~) Void
    +clearSession() Void
    +openSession(userID: Strin
g, sceneID: String) Void
}
```

Ciclo de Vida de uma Sessão de Analytics em EdKit



Como registrar Dados de Proveniência em um Modelo de Proveniência?

- Caso 1: Estrutura é representável com uma única representação de proveniência
  1. Use os métodos `this.AsActivity()`, `this.AsAgent()` ou `this.AsEntity()` para criar a representação
  2. Registre a representação no modelo de proveniência
- Caso 2: Estrutura requer 2 ou mais representações de proveniência e/ou depende de informações do modelo
  1. Crie múltiplas instâncias de `Activity`, `Agent` e/ou `Entity` para representar a estrutura
  2. Registre as instâncias no modelo de proveniência

Session: *Modelo de Proveniência* (`ProvenanceModel`) *que descreve uma sessão de jogo*

- Informações armazenadas incluem ID do usuário, ID da fase, data e hora da sessão e listas contendo as representações de proveniência registradas
    - Conteúdo é armazenado em um arquivo de log salvo localmente no sistema de arquivos

Como configurar uma sessão de Analytics em EdKit?

1. Crie um novo `GameObject` com um componente `SessionBehaviour` na cena
2. Inicie uma nova sessão usando o método `OpenSession`, passando identificadores de usuário e cena como parâmetros
3. Registre estruturas de proveniência usando o método `Register` no `SessionBehaviour`
4. Para encerrar a sessão, use o método `CloseSession`, passando o sumário da sessão como parâmetro

Detalhes Técnicos de `King`

Commands

`Command`: Estrutura de dados contendo informações sobre um comando

- Encapsula uma instância de `IPose`
- Pode ser comparada com outros comandos

Poses

Raise Left: *"Levante a mão esquerda"*

- Mão esquerda mais alta que a mão direita
- Pontuação é definida pela diferença na altura entre as mãos (max: 1)

Player

`PlayerData`: *Estrutura de Dados de Proveniência contendo o estado atual do jogador*

- Armazena a pontuação e o número de vidas do jogador
- Implementa métodos para criar as atividades a serem armazenadas no Analytics

`CommandSelector`: *Componente responsável por decidir qual é o próximo comando a ser realizado pelo jogador*

- Cuida da seleção do próximo comando e da validação de poses
- Seleção é aleatória, com uma configuração para evitar repetições caso desejado

`CommandRequester`: *Componente responsável por indicar ao jogador qual será o próximo comando*

- Dispara eventos de acordo com o comando requisitado

Raise Right: *"Levante a mão direita"*

- Mão direita mais alta que a mão esquerda
- Pontuação é definida pela diferença na altura entre as mãos (max: 1)

`PlayerLogic`: *Componente responsável por representar o jogador na cena*

- Encapsula uma instância de `PlayerData`
- Dispara eventos quando a pontuação muda, o jogador é atingido ou acabam o número de vidas
- Possui bindings para lidar com o início e o fim da sessão

`CommandEvaluator`: *Componente responsável por avaliar se a pose realizada pelo jogador está correta*

- Análise da pose ocorre após intervalos fixos de tempo
- Pode requisitar novos comandos ao seletor ou reenviar o comando atual

Hands Up: Pose para identificar quando ambas as mãos estão levantadas

- Pontuação é baseada nas seguintes afirmações
  - Mão esquerda acima do posicionamento da cabeça
  - Mão direita acima do posicionamento da cabeça

Tarefa

> *Testando o jogo com usuários, os mesmos constataram que o jogo está muito repetitivo e sem variedade e, como desenvolvedor, você ficou encarregado de implementar um novo comando para o jogo para treinar a habilidade de comandos cruzados.*
>
> *O novo comando tem o nome de "Bloqueio" e segue as seguintes regras:*
>
> - *Mão esquerda do jogador deve estar à sua direita*
> - *Mão direita do jogador deve estar à sua esquerda*
> - *Instrução verbal ao jogador: "Cross your hands by moving the left hand to the right and the right hand to the left"*
> - *Representação visual deve usar o material para Requisição ("Request").*
> - *Comando deve ser registrado pelo sistema de analytics do jogo como uma nova Activity do tipo "Block" que está relacionado ao Agent "Player"*

1. Adicionar o novo comando "Bloqueio”

1.1. Criar uma nova pose para EdKit cumprindo os requisitos acima

- Uma pose em EdKit pode ser criada com uma de três formas:

a. Criando um novo script com um tipo que implemente a `IPose`

- No script "Pose+Block.cs” com um novo tipo `BlockPose` que implementa a interface `IPose`
- No método "Evaluate”, crie um algoritmo para definir a regra de bloqueio usando a documentação do próprio método como guia.

b. Criando um novo script que cria uma instância de `Pose`

- No script "Pose+Block.cs”, crie uma nova variável estática "Block" em `PoseExamples`, que armazena uma instância de `Pose`

1.2. Adicionar o comando a lista de comandos identificáveis pelo jogo

- Vá ao arquivo "Command.cs” e adicione um comando "Block”, usando a pose previamente criada
- Vá ao arquivo "CommandSelectorLogic.cs” e adicione o comando à lista de comandos reconhecida pelo seletor
- Vá ao arquivo "CommandRequester.cs” e adicione a lógica necessária para executar o evento `onRequestBlock`

2. Adicionar Feedback para o comando

2.1. Criar uma nova Affordance para representar o novo comando

- No Editor, clique com o botão direito do mouse e vá em Create → EdKit → Affordance para criar uma nova affordance chamada “Block Request”
- Na Affordance, configure os elementos necessários para a requisição conforme à especificação da pose

2.2. Fazer o link do comando e sua Affordance com o sistema de requisição de comandos

- Na cena "KingDemo", procure pelo objeto "Commands” e no componente `CommandRequester`, configure os eventos para executar a nova Affordance

3. Registrar um evento de Analytics quando o comando for executado

3.1. Criar uma nova atividade para a pose

- Vá ao arquivo "PlayerData.cs” e crie uma novo método “SetBlock” para retornar uma atividade para o comando Block
- Vá ao arquivo "PlayerLogic.cs” e crie um novo método "Block” para criar a atividade e chamar o evento `onBlock`
- Vá ao arquivo "CommandEvaluator.cs” e modifique o método "EvaluateLastPose” para chamar o evento `onBlockPerformed`

3.2. Registrar a atividade na sessão toda vez que o comando for executado

- Na cena "KingDemo”, procure pelo objeto Player e, no evento `onBlock`, registre a atividade de bloqueio na Sessão de Analytics