MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Development of a Virtual Reality Game for Stroke Rehabilitation"

verfasst von / submitted by

Seyed Farhad Mohammadian Masouleh

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2025 / Vienna, 2025

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 926
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Wirtschaftsinformatik UG2002
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. -Ing.Moritz Grosse-Wentrup

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Prof. Moritz Grosse-Wentrup, for his invaluable guidance and support throughout my Master's thesis.

I would also like to thank Philipp Raggam for his helpful feedback and contributions during the research, Implementation, and writing process.

Abstract

Stroke rehabilitation is a critical process that requires intensive and repetitive training to restore motor functions. Traditional rehabilitation methods often face limitations in engagement and adaptability, leading to reduced patient motivation and suboptimal recovery outcomes. To address these challenges, this thesis presents a Virtual Reality (VR)-based rehabilitation system designed to enhance upper-limb motor recovery through interactive and immersive exercises. The system integrates real-time motion tracking and a physics-based interaction model to ensure natural hand-object interactions within a virtual environment.

A key feature of this system is the structured set of rehabilitation tasks that focus on hand coordination, precision, and functional movement. These tasks are designed to simulate real-world activities, providing patients with engaging and goal-oriented exercises. Additionally, the system incorporates real-time data collection and analysis via the Lab Streaming Layer (LSL) protocol, thereby allowing an objective assessment of user performance. Special calibration techniques are implemented to align virtual objects with real-world counterparts, ensuring accuracy in user interactions.

The system was tested and validated in a laboratory setting through a series of functional and usability tests, including assessments of motion tracking, calibration accuracy, and real-time data transmission. These evaluations demonstrated the feasibility and stability of the application. In addition, the system was tested with stroke patients by the Neuroinformatics research group, whose published results confirm its therapeutic potential, although the detailed outcomes are beyond the scope of this thesis. Compared to existing VR rehabilitation approaches, this work contributes a structured set of real-world–inspired tasks combined with real-time data collection through the LSL protocol and calibration techniques for precise alignment between virtual and physical environments.

The findings suggest that immersive VR-based rehabilitation can improve patient engagement and facilitate personalized and adaptable recovery processes. This research contributes to the growing field of digital rehabilitation by demonstrating how VR systems can be systematically integrated into stroke therapy, offering both clinical applicability and a foundation for future research on neurorehabilitation technologies.

Kurzfassung

Die Rehabilitation nach einem Schlaganfall erfordert intensive und wiederholte Trainingsmaßnahmen, um motorische Funktionen wiederherzustellen. Traditionelle Methoden stoßen jedoch häufig an ihre Grenzen, da sie wenig abwechslungsreich sind und dadurch die Motivation der Patienten verringern können. In dieser Arbeit wird daher ein Virtual-Reality-(VR)-basiertes Rehabilitationssystem vorgestellt, das gezielt die Wiederherstellung der motorischen Funktionen des Oberkörpers unterstützt. Das System kombiniert Echtzeit-Bewegungserfassung mit einem physikbasierten Interaktionsmodell, um natürliche Hand-Objekt-Interaktionen in einer virtuellen Umgebung zu ermöglichen.

Ein zentrales Merkmal ist die strukturierte Auswahl von Rehabilitationsaufgaben, die auf Koordination, Präzision und funktionelle Bewegungen abzielen. Diese Aufgaben orientieren sich an Alltagstätigkeiten und bieten den Patienten motivierende und zielgerichtete Übungen. Zusätzlich werden mithilfe des Lab Streaming Layer (LSL)-Protokolls Bewegungsdaten in Echtzeit erfasst und analysiert, wodurch eine objektive Leistungsbewertung möglich ist. Spezielle Kalibrierungsverfahren sorgen dafür, dass virtuelle Objekte präzise mit realen Gegenständen übereinstimmen.

Das System wurde im Labor durch eine Reihe funktionaler und nutzerorientierter Tests überprüft. Dabei wurden unter anderem die Genauigkeit der Bewegungserfassung, die Stabilität der Kalibrierung sowie die Zuverlässigkeit der Datenübertragung untersucht. Diese Tests belegen die technische Machbarkeit und Benutzerfreundlichkeit der Anwendung. Darüber hinaus wurde das System von der Neuroinformatics-Forschungsgruppe mit Schlaganfallpatienten erprobt. Die Ergebnisse dieser externen Studie liegen außerhalb des Rahmens dieser Arbeit, zeigen jedoch das therapeutische Potenzial des Systems.

Die Ergebnisse deuten darauf hin, dass VR-basierte Rehabilitation die Motivation der Patienten steigern und eine individuell anpassbare Therapieumgebung schaffen kann. Diese Arbeit leistet damit einen Beitrag zur digitalen Rehabilitation, indem sie aufzeigt, wie VR-Systeme systematisch in die Schlaganfalltherapie integriert werden können, und bildet eine Grundlage für zukünftige Forschungs- und Entwicklungsarbeiten im Bereich der Neurorehabilitation.

Contents

1 Introduction

1.1 Context and Motivation

Stroke accounts for 11% of all deaths worldwide, and approximately 30–40% of stroke survivors experience severe disabilities [1]. Consequently, stroke is one of the leading causes of long-term disability, often leaving patients with impaired motor function that affects their ability to perform daily tasks. Although traditional rehabilitation exercises can be effective, they are often repetitive and lack engagement, which reduces patient motivation and adherence to therapy. To address this challenge, the use of technology, particularly Virtual Reality (VR), has emerged as a promising tool in rehabilitation programs. VR offers an immersive and interactive environment where patients can engage in exercises that simulate real-world tasks, providing both a sense of achievement and progress. For instance, VR games create a safe and engaging environment for patients to practice hand exercises.

In VR games, patients use real-world simulations designed to improve hand coordination, strength, and fine motor skills. By combining the immersive nature of VR with game mechanics such as scoring, progress tracking, and performance feedback, the game motivates patients to participate more actively in their rehabilitation. VR games can provide patients with a dynamic, engaging alternative to traditional exercises. It is therefore expected that traditional rehabilitation methods will become less prominent in future visions of therapy, while VR games will establish themselves as a strong alternative. Thus, the main motivations of this work stem from the challenges of traditional rehabilitation methods in areas such as patient engagement, motor recovery, real-time feedback, data-driven personalization, providing a controlled, risk-free environment, and progress monitoring.

1.2 Challenges and Research Questions

In light of the motivation and described challenges, the underlying hypothesis of this thesis is as follows:

"Integrating real-time data collection and interactive tasks in a VR rehabilitation game can enhance motor function and patient engagement in stroke survivors."

The formulated hypothesis defines the research questions that form the basis of this thesis. This project aims to address these research questions to evaluate the hypothesis and then prove or reject it. The research questions are summarized as follows:

1. What are concrete tasks designed in the VR game to target different aspects of motor recovery in stroke rehabilitation, such as coordination, strength, and fine motor skills?
2. How to design a scoring system based on accuracy and speed serving as a motivational tool for stroke patients during their rehabilitation?
3. How to collect real data time in a VR game and transmit it to a central station?

1.3 Objectives

The main objectives of this thesis are summarized as follows:

- Development of a VR Rehabilitation Game: Create a virtual reality game using Unity and Oculus SDK that offers engaging and personalized rehabilitation tasks to improve motor skills and coordination for stroke survivors.
- User-Centric Design and Interaction: Design an immersive virtual environment that mimics a familiar home setting and provides users with intuitive interactions through augmented-reality hand representations, thereby promoting comfort and reducing anxiety during rehabilitation.
- Task-Oriented Rehabilitation: Implement a series of rehabilitation tasks focusing on essential motor functions, such as hand-eye coordination, grip control, and upper limb mobility, with features for repetition and feedback to enhance user performance and engagement.
- Real-Time Data Collection and Analysis: Integrate real-time data collection using Lab Streaming Layer (LSL) protocol to monitor user interactions, enabling dynamic assessment of motor skills and progress in rehabilitation, while correlating game performance with external physiological signals to study neuroplasticity.

Therefore, this thesis aims to develop an immersive VR game, called **Rehab App**, for post-stroke upper limb rehabilitation in a 3D virtual environment. Moreover, the developed Rehab App has been used in research conducted by the Neuroinformatics research group, presented at the 9th Graz Brain-Computer Interface Conference [2], demonstrating its potential early impact.

1.4 Research Methodology

This section outlines the research methodology employed in this thesis. The research methodology encompasses the following key steps:

- Literature Review: The first step involves a comprehensive literature review to gain insight into the current state of research regarding stroke rehabilitation through

VR. This review aims to provide an overview of existing methodologies, outcomes, and technological advancements in the field, including an examination of scholarly articles on VR rehabilitation, patient engagement, and motor function recovery. The literature review also identifies gaps in the existing research that this thesis aims to address. The findings from this review are detailed in Section 1.7.

- Design and Implementation: Following the literature review and requirement analysis, the next phase involves designing and implementing the VR rehabilitation game. This stage incorporates principles from user-centered design to ensure that the game is both engaging and effective for stroke survivors. The design process considers various interactive tasks and real-time feedback mechanisms aimed at enhancing patient engagement and promoting motor skill recovery. The implementation includes the development of the VR environment using Unity, integrating real-time data collection for assessing patient progress.
- Evaluation: The effectiveness of the VR rehabilitation game is assessed through a series of experiments. A set of unit tests is conducted to ensure the reliability and functionality of individual components within the VR game. These tests evaluate critical aspects such as motion tracking accuracy, real-time data collection, task logic, and the stability of the VR environment. In addition, a pilot study is conducted by the Neuroinformatics research group with a small group of stroke survivors to evaluate the initial functionality, usability, and therapeutic potential of the VR game. Engagement is evaluated through both quantitative (e.g., time spent on tasks) and qualitative measures (e.g., participant feedback by the Neuroinformatics research group).

1.5 Structure of the Thesis

This section outlines the organization of this thesis. The following subsections introduce the preliminary concepts that form its foundation of this thesis, covering key topics including: VR, VR Devices, VR Therapy, and VR Gaming Development. Section 1.7 delves into related work, comprehensively reviewing existing literature and relevant studies. Chapter 2 details the Methods employed in this research, outlining the design, development, and evaluation processes of the VR rehabilitation game. Chapter 3 presents the Results obtained from the experimental studies, highlighting the outcomes related to motor function recovery and patient engagement. Chapter 4 provides a Discussion of the results, interpreting the findings in the context of existing literature and examining their implications for stroke rehabilitation. Finally, Chapter 5 concludes the thesis, summarizing the key findings and outlining future directions for research in VR rehabilitation.

1.6 Preliminaries

1.6.1 Virtual Reality

The concept of VR was first introduced in the 1960s, with Morton's creation of the Telesphere Mask and the Sensorama [3]. VR is a technology that uses computer-generated environments to simulate a user's physical presence in a three-dimensional, interactive virtual world [4]. VR typically involves wearing a headset that tracks the movement of the user's head and, in some cases, their hand movements as well. This allows the user to look around and interact with the virtual environment as if they were truly present in that space.

VR aims to create a sense of presence and immersion by stimulating multiple senses, primarily vision and sometimes hearing and touch. To achieve this, VR technology often employs high-resolution displays, motion tracking sensors, and spatial audio to create a convincing virtual experience. This technology is used for various purposes such as gaming [5, 6], training and simulation [7, 8], healthcare, and therapy.

1.6.2 Virtual Reality Devices

Several VR devices are available on the market, ranging from high-end systems to more accessible and affordable options. These devices offer varying levels of immersion, comfort, and capabilities. Here are some of the prominent VR devices available during the period in which this study was conducted:

1. Oculus Quest 2: Oculus Quest 2 is a standalone VR headset developed by Facebook's Oculus division. It offers both wireless and PC-tethered VR experiences. When tethered to a compatible PC, it can provide more graphically intensive experiences. It features inside-out tracking, which means it uses built-in cameras to track the user's movements without external sensors. It's known for its affordability, ease of use, and a growing library of games and applications.

2. PlayStation VR (PS VR): Developed by Sony, PS VR is a VR headset designed for the PlayStation 4 and PlayStation 5 gaming consoles. It offers various gaming experiences, taking advantage of the PlayStation ecosystem. It requires a PlayStation console to function and offers a variety of games and experiences through the PlayStation Store.

3. Pico Neo 2: The Pico Neo 2 is a standalone VR headset designed for enterprise and professional use. It features 6DoF (six degrees of freedom) tracking and is often used for training, simulations, and industrial applications.

4. Valve Index: The Valve Index is a high-end PC VR headset developed by Valve Corporation. It offers precise tracking and finger-tracking controllers, allowing for more realistic hand interactions. It's designed for use with powerful gaming PCs and is known for its high refresh rates and impressive graphics.

5. Windows Mixed Reality (WMR) Headsets: These headsets are developed by various manufacturers, including HP, Acer, Lenovo, Samsung, and more. They offer a range of features and price points, with inside-out tracking and compatibility with Windows PCs.

6. HTC Vive Series: HTC offers a range of VR headsets, including the original Vive, Vive Pro, and Vive Cosmos. The Vive Pro offers higher resolution and enhanced comfort, while the Vive Cosmos features inside-out tracking and a modular design. These headsets often come with controllers and external sensors for room-scale tracking.

To conduct this thesis, the Oculus Quest 2 was selected for several key reasons:

- Affordability: The Oculus Quest 2 is priced lower than other high-end VR systems, making it a cost-effective option.
- Standalone Functionality: As a standalone VR headset, it does not require a powerful PC or console to operate, enhancing portability and making setup easier in different environments.
- Wireless Freedom: The wireless nature of the Quest 2 enables untethered movement, increasing immersion and allowing for enhanced physical interaction, particularly in tasks involving significant motion.
- Inside-Out Tracking: The Quest 2 utilizes built-in cameras for inside-out tracking, eliminating the need for external sensors and simplifying the setup process.
- Ease of Setup: Designed with ease of use in mind, the Quest 2 features a quick and straightforward setup, making it accessible to non-technical users without the need for external sensors or complex calibration.
- Expanding Content Library: The Oculus Quest 2 has access to a rapidly expanding library of VR applications and games through the Oculus Store, providing a wide range of experiences suited to diverse tasks and experiments.
- Oculus Link Compatibility: With the Oculus Link cable, the Quest 2 can be connected to a compatible gaming PC, enabling access to more graphically intensive PC VR content, and offering flexibility between standalone and PC-based experiences.
- Comfortable Design: The Quest 2 features an ergonomic design with adjustable straps and a lighter form factor, making it more comfortable for extended use.

Although the Oculus Quest 2 has certain limitations, such as privacy concerns, battery life, IPD adjustment, and the need for Guardian area setup, the advantages mentioned above make it the ideal choice for this project.

1.6.3 Virtual Reality Therapy

VR has become a powerful tool for medical education, therapy, and rehabilitation due to its immersive nature and ability to create controlled and customizable environments. For example, it allows patients to project themselves and their movements into simulated environments and facilitates motor rehabilitation through engaging and interactive exercises. Through the use of playful activities, VR offers intensive motor training in an entertaining way, while progress can be tracked at the same time. This approach enables measurable insights into rehabilitation progress and provides patients with valuable feedback on their performance, which is crucial for motor learning after a stroke.

The increasing number of stroke survivors has led to a greater demand for rehabilitation services. Stroke often leads to multiple impairments, including motor, cognitive, and emotional deficits. It is estimated that nearly 1% of the world's population lives with the consequences of a cerebrovascular event [9][10]. Among these survivors, chronic motor disabilities such as hemiplegia and hemiparesis are the most common and debilitating consequences [10]. Consequently, most rehabilitation efforts are focused on restoring motor function, emphasizing the need for innovative and efficient interventions.

Stroke survivors often face the challenge of relearning their motor skills, regaining their balance, and restoring their independence in everyday activities. VR provides a dynamic platform that bridges the physical and virtual worlds, allowing patients to immerse themselves in interactive exercises that target specific motor and cognitive functions. By simulating real-life scenarios in controlled environments, VR encourages repetitive and purposeful movements, promoting the reorganization of neural pathways and neuroplasticity.

Numerous studies have shown that VR can significantly improve motor function after a stroke [11]. Recent research suggests that VR is a valuable addition to traditional upper limb rehabilitation treatments. However, the widespread use of VR therapy is limited due to the high cost of specialized equipment and the need for trained operators.

A more accessible alternative to customized VR therapy is the use of commercial video game systems such as the Nintendo Wii and Oculus Quest for post-stroke rehabilitation. These systems have been evaluated for upper limb rehabilitation and have shown promising results in improving motor function while increasing patient satisfaction and enjoyment[12][13].

One of the key advantages of VR-based rehabilitation is its ability to tailor exercises to each patient's unique needs and capabilities. Customized virtual environments can replicate activities of daily living, enabling stroke survivors to practice tasks that align with their individual goals. These exercises challenge patients to improve their motor skills, coordination, and overall functionality in an engaging and supportive manner.

Game-based activities in VR deliver intensive motor training while offering real-time feedback mechanisms. This feedback provides stroke survivors with immediate insights into their performance during exercises, fostering self-awareness and empowering them to make necessary adjustments for more accurate and effective movements. As stroke recovery is a gradual process, visual and auditory cues within VR environments serve as motivational tools by showcasing progress over time, instilling a sense of achievement, and maintaining momentum.

VR technology represents a groundbreaking advancement in stroke rehabilitation. By combining immersive environments, tailored exercises, and real-time feedback, VR addresses the multifaceted challenges of recovery. It offers innovative and accessible solutions that improve motor function, enhance patient engagement, and transform rehabilitation practices.

1.6.4 VR Gaming Development

To develop VR games, developers should use game engines that fully support VR technology. Among the available options, Unity, Unreal Engine, and CryEngine are prominent choices [14].

In this work, Unity is selected for several key reasons. First, it is known for its ease of use and accessibility, particularly benefiting beginners. Unity's primary programming language, C#, is versatile, user-friendly, and object-oriented. Additionally, Unity offers a vast library of community-shared assets, which simplifies development by providing pre-built components [14]. Moreover, Unity's broad platform compatibility allows it to support a range of VR devices, including Oculus Quest 2.

The fundamental Unity concepts are summarized as follows:

- Game Objects: Game Objects are the basic building blocks in Unity representing elements in a game, such as characters, props, lights, cameras, etc.
- Components: Components are used to add functionality to Game Objects. Depending on the complexity of the object, a combination of components can be used to implement the desired functionality. The main components in Unity are summarized as follows:
  - Transform: The Transform component defines the position, rotation, and scale of a Game Object in the 3D world.
  - Renderer: Components like Mesh Renderer, Line Renderer, and Trail Renderer are responsible for rendering the visual representation of Game Objects.
  - Collider: Colliders are used to define the physical shape of a Game Object for the purpose of collision detection.
  - Rigidbody: The Rigidbody component is used to apply physics forces to a Game Object. It enables objects to respond to gravity, forces, and collisions.
  - Audio Source: This component is used to play audio clips and manage various audio properties like volume, pitch, and spatial blend.
  - Light: Unity provides various types of lights (Directional Light, Point Light, Spot Light) to illuminate the scene.
  - Camera: The Camera component defines the viewpoint of the player or the viewpoint from which the scene is rendered.

  - Script (MonoBehaviour): This is one of the most important components. It allows you to attach custom scripts written in C# to define the behavior of a Game Object.
  - Animator: The Animator component is used for controlling animations. It works in conjunction with an Animator Controller.
  - Particle System: The Particle System component is used to create and control effects like fire, smoke, rain, and more.
  - Canvas and UI Components: Canvas is used to render UI elements. Components like Text, Image, Button, etc., are used to create user interfaces.
  - Terrain: The Terrain component is used to create large outdoor environments with features like hills, valleys, and foliage.
  - Terrain Collider: Specifically designed for terrains, this collider allows for more efficient collision detection with complex terrain surfaces.

- Scenes: A Scene is a fundamental concept that represents a specific level, environment, or set of game elements. Each scene can contain multiple Game Objects.
- Asset Store: In Unity, assets are the items and objects that can be used in the project. The Unity Asset Store is a marketplace where pre-made assets, scripts, plugins, and more can be found to speed up the development process.

1.7 Literature Review

Studies have demonstrated that if significant motor function loss persists beyond the initial weeks post-stroke, the probability of recovering functional abilities decreases sharply over time [15, 16], underscoring the critical need for innovative rehabilitation approaches to enhance recovery outcomes in such cases. Recently, VR has emerged as a promising technology, offering beneficial characteristics for the functional recovery of stroke patients [17]. Over the past decade, numerous studies have consistently proven the effectiveness of VR games [18, 19, 20].

Compared to traditional rehabilitation approaches (i.e., those provided by physical and occupational therapists in clinical settings, including activities of daily living training), VR games offer real-time feedback on movement features, which, according to neuroscience studies, enhances the ability to learn and perform tasks more effectively. Compared to conventional therapy (CT) delivered by physical and occupational therapists, VR is currently employed as a complement to CT in clinical settings [1].

Several VR games, such as Butterfly, Space War, Snow, Tubes, and Virtual Home, have been developed, each featuring different movement types designed to support rehabilitation and therapeutic goals [1]. As reviewed in [1], in the Butterfly game, the patient is tasked with scaring off butterflies using hand movements. The exercises involve flexion and extension of the elbow with supination, as well as horizontal shoulder movements such as turning and bending. In the Space War game, the player guides a spacecraft and prevents collisions with meteorites. The required movements include shoulder flexion and extension with elbow extension, as well as horizontal shoulder adduction and abduction with the elbow in an open position. The objective of the Snow game is for the patient to collect snowflakes by reaching out their hand. The movements primarily involve shoulder flexion and adduction, elbow bending and stretching, and horizontal shoulder movements with elbow supination. In the Tubes game, the patient directs a green object through three tubes, causing the tubes to disappear when the object reaches the end. The exercises include elbow flexion and extension, elbow supination, and horizontal shoulder flexion and extension. The Virtual Home game simulates household tasks, such as moving pillows, making coffee, closing drawers, and setting dishes. The movements required for these tasks include upper limb elevation with internal and external shoulder rotation, shoulder adduction and abduction, and shoulder flexion. Moreover, these VR games differ in terms of portability of the system (e.g., whether they can be used at home or in clinical environments), number of players, user interface (2D or 3D), mechanism of interaction (e.g., text, sound, or score), and software.

A VR rehabilitation game called Motion Rehab AVE 3D is developed in [21] to enhance motivation in patients undergoing upper limb and balance training after a mild stroke. In this game, patients stand and stretch their shoulders, elbows, and wrists to catch falling objects and earn points, with movements aligned to traditional rehabilitation exercises. Although the game provides a training experience, it lacks a meaningful system for using earned points and does not include an integrated storyline.

The games developed in [22, 23] are designed to enhance upper-limb movement in stroke patients. The former uses a mirrored effect to map movements from the healthy limb to the affected one, enabling patients to stack stones and arrange items in a virtual kitchen, with therapists adjusting task difficulty as needed [22]. The latter game also applies mirror therapy to promote bilateral coordination by having patients pick up and move virtual objects with mirrored actions [23]. However, both games lack engaging content to maintain patient interest for long-term training.

Therefore, although multiple research studies have focused on developing specific VR games for stroke rehabilitation, there is still significant potential for immersive VR-based designs aimed at enhancing stroke rehabilitation [24]. In contrast to the VR games mentioned above, the Rehab App game developed in this thesis is specifically designed to engage a wide range of upper-limb movements. It emphasizes sustained patient engagement by incorporating complex combinations of flexion, extension, rotation, and stabilization to support functional motor recovery. Additionally, it utilizes real-time data analysis collected during gameplay to enhance the rehabilitation experience and interactions.

2 Methods

2.1 Conceptual Overview of the Game

The Rehab App was developed as part of a stroke rehabilitation program, specifically targeting hand exercises to assist patients in recovering motor function. The game aims to assist patients in regaining motor function through a series of interactive tasks that simulate real-world movements and challenges. The core purpose of the game is to provide patients with engaging, repetitive exercises that help improve coordination, strength, and fine motor skills, while also tracking their progress over time. The game consists of three distinct tasks, each tailored to target different aspects of motor recovery. As patients complete these tasks, the system tracks their performance, awarding points based on accuracy and speed. These points serve both as a motivational tool and as a measurable indicator of the patient's progress during rehabilitation

In addition to providing immediate feedback to the patient, the game collects detailed performance data, which is transmitted to a central computer for further analysis by healthcare professionals. This data includes metrics such as event markers, movement precision, task completion time, and error rates, which can be used to adjust the therapy regimen and assess long-term improvement. The use of VR in this context allows for an immersive and controlled environment where patients can safely practice rehabilitation exercises with real-time feedback and monitoring.

2.2 Technical Overview of the Game

The Rehab App was developed using Unity, leveraging its built-in physics engine, asset management system, and support for VR development through Oculus SDK. The core architecture follows a modular approach in which each scene of the game represents a distinct level or challenge. Scripts written in C# manage player interactions, object physics, and game logic. A central game manager script oversees transitions between scenes and handles global game states, such as player health, progress, and score.

2.3 Game Workflow and Interaction Design

The virtual environment of the game is designed to create an engaging and comfortable space that facilitates the rehabilitation tasks for stroke survivors. This environment is a key element of the game, as it not only supports the gameplay but also enhances the

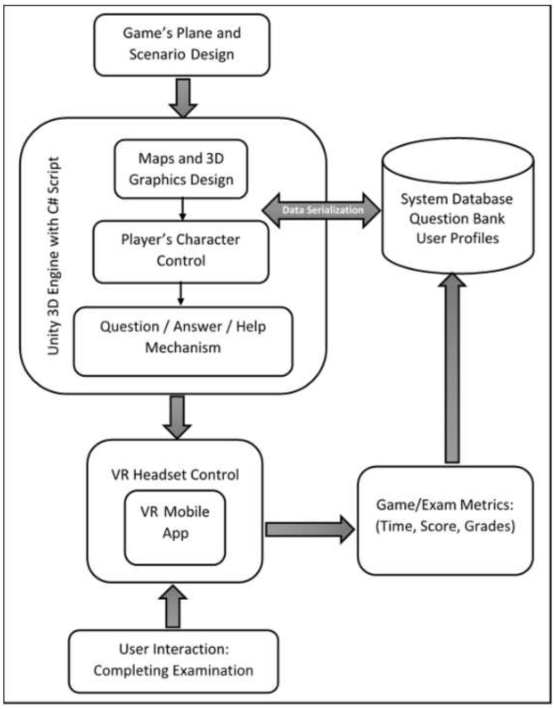

Figure 2.1: VR Game Diagram

user's immersion and motivation throughout the rehabilitation process. The workflow and interaction design are structured into the following parts: Room Design and Setup, VR Immersion, Canvas Menus, Options Menu, and Result Canvas Menu.

2.3.1 Room Design and Setup

The room is furnished with essential elements like couches, windows, a shelf, and plants. These elements are meant to enhance the room's coziness, mimicking a familiar home environment, which helps reduce the user's stress and anxiety during rehabilitation. The plants are placed strategically to add a natural element, promoting a sense of tranquility and well-being. These plants are decorative but contribute to the immersive and peaceful atmosphere of the space.

There is a table and a chair, which represent the starting point for the user. These objects are placed at the center of the environment, emphasizing their functional role in the tasks. The user remains seated at the table throughout the game, where all tasks are performed. The table acts as the primary interaction surface for the user, with virtual objects appearing on it during tasks.

2.3.2 VR Immersion

The user is fully immersed in the virtual room through the VR headset, providing a 360-degree view of the surroundings. However, the user's interaction space is limited to the immediate area around the table. A key feature of the game is the inclusion of Augmented Reality (AR) representations of the user's hands. The user can see their hands projected virtually on the screen, allowing them to interact with objects on the table. This AR feature enables real-time feedback, allowing users to perform tasks as if they were using their actual hands, which is vital for the rehabilitation of the motor skill. The AR hands closely mimic the movements of the user's real hands, giving them control over virtual objects and enhancing the sense of agency and engagement in the rehabilitation process.

2.3.3 Canvas Menus

Several Canvas Menus are presented to the user, designed to be intuitive and accessible to stroke survivors who may have limited mobility or cognitive difficulties. Upon launching the game, the user is presented with the Main Menu, as illustrated in Figure 2.2. The Main Menu provides two primary options. The user can modify game options and configuration by selecting the Options button. To start the game, the user can select the Start button, which initiates a series of rehabilitation tasks.

Figure 2.2: Main Menu

In the game scripts, a CanvasManager.cs script was implemented to manage the visibility of different canvas menus during the user's interactions. This script allows the game to switch between various canvases, such as the Main Menu, Options Menu, Results Panel, etc. Below is a code snippet showing how canvases are shown and hidden using this script.

ShowCanvas methods are responsible for displaying a specific canvas based on its type or given CanvasController object. The current active canvas is stored in activeCanvas property during the game. Before displaying the new canvas, it hides the currently active canvas, then makes the target canvas visible and updates the active canvas reference. Both methods ensure that only one canvas is visible at a time and handle any errors gracefully by logging warnings.

```
public void ShowCanvas(CanvasType canvasType)
{
    CanvasController canvas = GetCanvasController(canvasType);

    if (canvas == null || canvasType.Equals(CanvasType.Default))
    {
        Debug.LogWarning(string.Format("CanvasManager - error displaying
    canvas. Canvas not found: {0}.", canvasType));
        return;
    }

    HideCanvas(activeCanvas);

    canvas.gameObject.SetActive(true);
    activeCanvas = canvas;
}

public void ShowCanvas(CanvasController canvas)
{
    if (canvas == null || canvas.canvasType.Equals(CanvasType.Default))
    {
        Debug.LogWarning(string.Format("CanvasManager - error displaying
    canvas. Canvas not found: {0}.", canvas));
        return;
    }

    HideCanvas(activeCanvas);

    canvas.gameObject.SetActive(true);
    activeCanvas = canvas;
}
```

Listing 2.1: CanvasManager: Show Canvases

The HideCanvas method sets the active canvas reference to null. It hides the active canvas.

```
public void HideCanvas(CanvasController canvas)
{
    if (canvas != null && canvas.isActiveAndEnabled)
    {
        canvas.gameObject.SetActive(false);

        if (canvas.Equals(activeCanvas))
        {
```

```
            previousCanvas = activeCanvas;
            activeCanvas = null;
        }
    }
}
```

Listing 2.2: CanvasManager: Hide Active Canvase

2.3.4 Options Menu

The Options Menu, shown in Figure 2.3, provides a set of configurable settings that allow users or therapists to tailor the game to individual rehabilitation needs. This menu is designed to ensure the user can optimize the game environment and tasks for effective and personalized rehabilitation. The options are presented in an easy-to-navigate interface, allowing users to adjust settings using AR hands before starting the tasks.

Figure 2.3: Options Menu

The user can randomize the order of the rehabilitation tasks. By enabling task randomization, the game introduces variety into the session, preventing repetition and increasing engagement. This also helps with cognitive flexibility, as the user cannot predict the next task.

The game offers an option to configure the settings for left-handed users. This feature ensures that all tasks (especially in Task 2) are adapted for individuals who use their left hand as the primary hand during rehabilitation. The objects are mirrored to accommodate left-hand dominance, making the experience more accessible for left-handed stroke survivors.

Before starting the game, it is important to ensure that the virtual table in the game aligns properly with the user's real-world physical table. This calibration is crucial for ensuring that the virtual table and the AR hands match the user's actual arm position and movement range. It prevents any discrepancies between the virtual and real-world coordinates, improving the accuracy and effectiveness of the tasks. Calibration typically involves adjusting the height and position of the virtual table to match the user's seated position.

Under session settings, users can adjust several parameters related to the rehabilitation session like the number of tries per Task. This setting determines how many attempts the user will have for each task. For example, the therapist or user can configure the number of tries based on the difficulty level or fatigue concerns.

In the game scripts, the SettingsManager.cs class likely serves as a centralized place to manage the saving and loading of various game and task settings, using Unity's PlayerPrefs to persist these settings between play sessions. Some related settings are also defined in the TaskSettings class located within the Task.cs script and manages settings specific to tasks like the number of tries (taskDuration property). Other settings, such as current duration and current score, are defined within the game and are not user-configurable.

```
[System.Serializable]
public class UserSettings
{
    public float maxReachUpward = 0;
    public float maxReachForward = 0;
    public string randomTasks = "false";
    public string leftHanded = "false";
    public float OffsetX = 0.0f;
    public float OffsetY = 0.0f;
}
```

Listing 2.3: SettingsManager: User Configurations

```
[System.Serializable]
public class TaskSettings
{
    public int taskDuration = 5;
    public int currentScore = 0;
    public int currentDuration = 0;
    public Difficulty difficulty = 0;
}
```

Listing 2.4: SettingsManager: Tasks Configurations

The SettingsManager class contains the SaveSettings method, which stores both task-specific settings and user settings back into PlayerPrefs in the form of serialized JSON strings. The method serializes the task's settings into JSON, then stores it in PlayerPrefs with separate keys for each User and task settings.

```
public void SaveSettings()
{
    foreach (Task t in GameManager.Instance.Tasks)
    {
        if (t.Settings != null)
        {
            var json = JsonUtility.ToJson(t.Settings);
            PlayerPrefs.SetString(t.Type.ToString(), json);
        }
    }

    // save user settings
    var json2 = JsonUtility.ToJson(Settings);
    PlayerPrefs.SetString(settings_JSON, json2);
}
```

Listing 2.5: SettingsManager: Tasks Configurations

The LoadSettings method in SettingsManager class retrieves saved data for both task-specific settings and user settings from PlayerPrefs, a lightweight way to store and load settings persistently. It sets task settings by loading the data from PlayerPrefs and deserializes the task-specific settings (like number of tries, duration, etc.) into the TaskSettings class. It also retrieves the JSON string and deserializes it into an instance of UserSettings. If the settings are missing or fail to load, a new UserSettings object is initialized with default values.

```
public void LoadSettings()
{
    // load task settings
    foreach (Task t in GameManager.Instance.Tasks)
    {
        t.Settings = LoadTaskSettings(t);
    }

    // load user settings
    if (PlayerPrefs.HasKey(settings_JSON))
    {
        Settings = JsonUtility.FromJson<UserSettings>(PlayerPrefs.
    GetString(settings_JSON));
        if(settings == null)
        {
            settings = new UserSettings();
            var json2 = JsonUtility.ToJson(settings);
            PlayerPrefs.SetString(settings_JSON, json2);
        }
    }
    else
    {
```

```
        // load default settings
        Settings = new UserSettings();
    }
    TextLeftHanded.text = settings.leftHanded.ToUpper();
    TextRandomTasks.text = settings.randomTasks.ToUpper();
}
```

Listing 2.6: SettingsManager: Tasks Configurations

In the SettingsManager class, there are several methods like ToggleLeftHanded, ToggleRandomTasks, SetOffsetX, SetOffsetY, and SetTaskDuration that are responsible for updating the settings based on user input through the game's Setting Canvas and Calibration Canvas. These methods modify values in the UserSettings and TaskSettings objects and save the changes accordingly.

The CalibrationManager will be integrated with the Options Menu to allow users to initiate and adjust the calibration process during gameplay. From the Options Menu, the user can trigger the calibration process to ensure the alignment of their virtual hands and the virtual table with their real-world counterparts. This class calls the SetOffsetX and SetOffsetY methods from SettingsManager class to adjust the offset values of the virtual table in relation to the user's starting position. This adjustment is crucial in ensuring that users have an accurate and comfortable alignment of their hands with the virtual objects on the table. The offsets accommodate variations in user height, seating position, or any physical constraints that may affect their interaction with the game. In the game, certain settings, such as Camera rotation and Table edge calibration, will be treated as transient settings, meaning they will not be permanently saved between sessions. These transient settings are likely intended for adjustments during a single play session and reset to default values when the game restarts.

```
public void RotateCameraRigLeft()
{
    Vector3 rot = cameraRig.transform.rotation.eulerAngles;
    cameraRig.transform.eulerAngles = new Vector3(rot.x, rot.y + 1, rot.z
    );
}

public void RotateCameraRigRight()
{
    Vector3 rot = cameraRig.transform.rotation.eulerAngles;
    cameraRig.transform.eulerAngles = new Vector3(rot.x, rot.y - 1, rot.z
    );
}

// save farthest hand position in world space placed on the desk. If both
    hands are tracked, pick the reach of the hand reaching closer. Called
    through UI.
public void SetMaxReachForward()
{
    Vector3 rot = cameraRig.transform.rotation.eulerAngles;

```

```
    float calibCameraY = rightHandReach.transform.position.y;
    float calibCameraX = rightHandReach.transform.position.x;

    var tablePos = table.transform.position;
    Vector3 camPos = cameraRig.transform.position;

    var diff_x = rightHandReach.transform.position.x - tablePos.x -
   TableWidth;
    var diff_y = rightHandReach.transform.position.y - TableHeight +
   FingerHeight;

    cameraRig.transform.position = new Vector3(camPos.x - diff_x, camPos.
   y - diff_y, camPos.z);
}
```

Listing 2.7: SettingsManager: Tasks Configurations

2.3.5 Result Canvas Menu

After completing all three tasks, the user is presented with a Result Canvas Menu. It shows a breakdown of the user's performance across the tasks, showing metrics such as time taken, accuracy, and number of successful completions, which can play a crucial role in motivating users and guiding their rehabilitation progress. There are some options to either repeat the tasks, adjust settings, or exit the game. The user can return to the Start Menu from the Result Canvas Menu or choose to end the game.

2.4 Tasks Description

The game consists of a series of rehabilitation tasks designed to improve motor function, coordination, and control, crucial for stroke recovery. Each task in the game is performed through multiple tries, where the user repeats the same action several times within the task session. Repetition is the key to reinforcing motor skills and allowing the user to progressively refine their movements.

The GameManager class serves as the core control system of the game, orchestrating the overall flow of the rehabilitation experience by managing sessions and tasks. It is responsible for transitioning between various game states and maintaining the structure of gameplay. It controls the overall flow of the game session, including starting, ending, and resetting sessions, and maintains the session state to ensure that tasks are executed in the correct order. This class monitors and responds to events such as scoring and hand position to handle task flow. The LoadNextTask method serves as a control point for managing the progression through the tasks. Depending on user settings (e.g., whether tasks should be randomized), it will either select the next task in the predefined order or shuffle the tasks for a more dynamic experience.

```
public void LoadNextTask()
{
    if (RandomTask)
    {
        System.Random rnd = new System.Random(DateTime.Now.Millisecond);
        int rndTask = rnd.Next(0, 3);

        int i = 0;
        while (finishedTasks[rndTask])
        {
            if (IsLastTask())
            {
                EndSession();
                return;
            }
            rndTask = rnd.Next(0, 3);
        }

        finishedTasks[rndTask] = true;
        EndTask();
        StartTask(tasks[rndTask]);
    }
    else
    {
        // load first task if no task active
        if (currentTask == null)
        {
            StartTask(tasks[0]);
        }

        // end session after last task
        else if (IsLastTask())
        {
            EndSession();
        }
        else
        {
            // else load next task in order
            EndTask();
            StartTask(tasks[tasks.IndexOf(currentTask) + 1]);
        }
    }
}
```

Listing 2.8: SettingsManager: Tasks Configurations

Before beginning a task—or starting a new try after completing one—the user is required to place their hands in a specific area on the virtual table which is highlighted visually, This hand placement serves as a calibration step, confirming that the user is properly aligned with the virtual environment and prepared to perform the task or next try. At the start of any task, once the hands are positioned correctly, a countdown begins from 3, giving the user a brief moment to prepare for the task ahead. After the countdown reaches zero, the task begins.

The Task class serves as a fundamental component in the game, acting as the top-level structure that manages all aspects related to specific tasks within the rehabilitation experience. This class is responsible for loading resources, managing task objects, and ensuring that tasks are set up correctly for various game states.

```
private void LoadResources(bool value)
{
    foreach (GameObject go in taskResources)
    {
        go.SetActive(value);

        foreach (Transform child in go.transform)
        {
            child.gameObject.SetActive(value);
        }
    }
}
```

Listing 2.9: SettingsManager: Tasks Configurations

Once a task or the allocated number of tries is completed, the game provides immediate feedback as a dialog box in front of the user, as shown in Figure 2.4. In the dialog box, two buttons are provided to navigate to the next task or exit the session.

Figure 2.4: Task Result Canvas

2.4.1 Task 1: Pouring Water

The first task in the sequence involves a common household action: pouring water into a glass. This task focuses on enhancing hand-eye coordination, grip control, and bilateral motor function, all of which are critical for stroke recovery. The user picks up the bottle and the glass at the first step. Using their AR hands, the user must grab the bottle and the glass, each with one hand. Then user carefully tilts the bottle to pour water into the glass and fill it. This action requires precise hand movements and control to avoid spilling the water, promoting smooth motor coordination. In the end, the user must place the bottle and the glass back onto the table in their original positions, ensuring proper object manipulation and placement, and put their hand on the table to prepare objects to start the next try. This task mimics real-life activities and encourages the user to practice fine motor skills and dual-hand coordination, which are often affected by stroke.

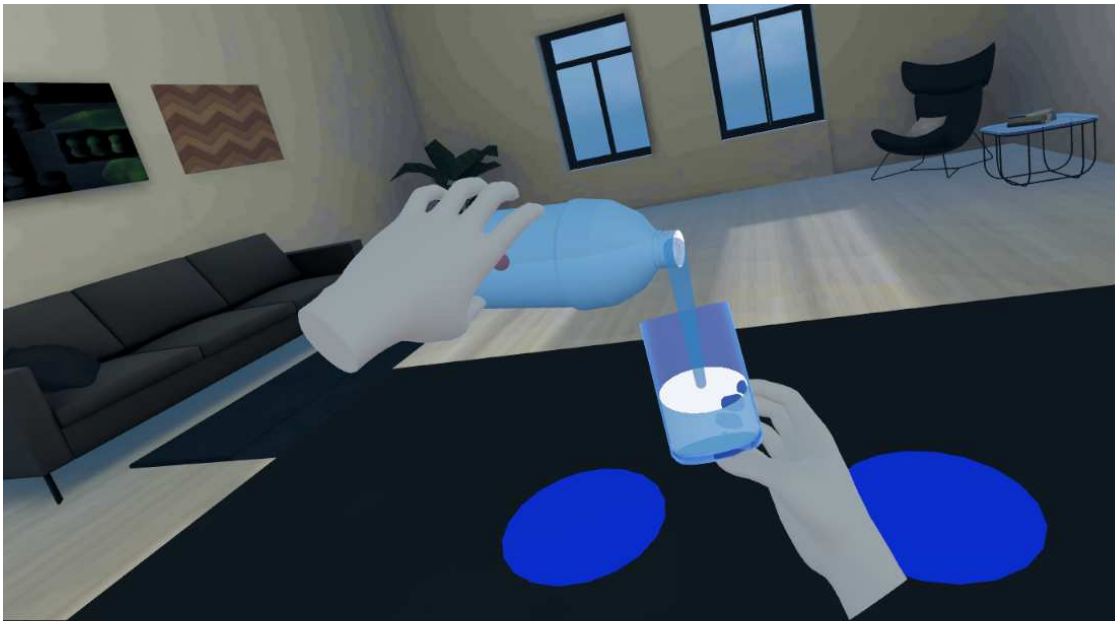

Figure 2.5: Task 1 - Pouring Water

A Task1 class is defined to handle the logic of this task. It consisted of an Update() method that plays a crucial role in maintaining the game's visual integrity and ensuring accurate scoring by continuously monitoring the position and state of the glass and bottle on the table. This real-time update allows the game to evaluate the correctness of task completion and the user's performance, providing feedback based on their actions, while also checking if the task of pouring water into the glass and placing both objects back in the correct position on the table is successfully completed. The method continuously checks the position of the glass and bottle, and every time the user interacts with them, the method is triggered to validate object positioning and task correctness to ensure real-time feedback and updates.

```
public void Update()
{
    ...

```

```
    // minimum height of the glass (glass not throw the Table)
    minYLevelofGlass = getMinYOfGlass(rotGlass);
    bool isOnTableArea = isObjectOnTableArea(posGlass);
    if (posGlass.y < minYLevelofGlass &&
        posGlass.y > 0.5f &&
        glassGrabbable.IsHeld &&
        isOnTableArea)
    {
        glassGrabbable.ResetPositionY(minYLevelofGlass);
    }

    float minYLevelOfBottle = getMinYOfBottle(rotBottle);
    // minimum height of the bottle
    ...

    glassOnTable = (Mathf.Abs(posGlass.y - minYLevelofGlass) < 0.01);
    bool bottleOnTable = (Mathf.Abs(posBottle.y - minYLevelOfBottle) < 0.1);

    if(glassOnTable && !scored && glass.filled > lastGlassFilled)
    {
        glass.filled = lastGlassFilled;
    }
    lastGlassFilled = glass.filled;

    bool objectsOnTheTable = glassOnTable && !glassGrabbable.IsHeld && bottleOnTable & !bottleGrabbable.IsHeld;

    if (glass.filled == 0)
        scored = false;

    //task Done Checking
    if (objectsOnTheTable)
    {
        if (glass.filled >= glass.maxCapacity * 0.9)
        {
            bool glassPutDownOnArea = (distanceXZ(posGlass, putDownGlass1.transform.position) < 0.07);
            bool bottlePutDownOnArea = (distanceXZ(posBottle, putDownBottle.transform.position) < 0.07);

            bool isGlassStanding = true || (rotGlass.x > 0.68 && rotGlass.x < 0.72) || (rotGlass.x < -0.68 && rotGlass.x > -0.72);
            bool isBottleStanding = true || (rotBottle.x > 0.68 && rotBottle.x < 0.72) || (rotBottle.x < -0.68 && rotBottle.x > -0.72);

            bool successfulGlassOnArea = glassPutDownOnArea && !glassGrabbable.IsHeld && isGlassStanding;
            bool successfulBottleOnArea = bottlePutDownOnArea && !bottleGrabbable.IsHeld && isBottleStanding;

            bool done = successfulGlassOnArea && successfulBottleOnArea;
            if (done && !scored)
```

```
            {
                scored = true;
                //glass.filled = 0;
                ScoreManager.Instance.IncreaseScore(task, 1);
                LSLSender.SendLsl("Task 1 - Put on Area", new float[] {
    130 });
            }
        }
    }

    //drop the glass
    if (!glassGrabbable.IsHeld && !glassOnTable)
    {
        if (!dropGlass)
            LSLSender.SendLsl("Task 1 - Drop the Glass", new float[] {
    141 });

        dropGlass = true;
    }
    if (glassGrabbable.IsHeld && !glassOnTable) dropGlass = false;

    //drop the bottle
    if (!bottleGrabbable.IsHeld && !bottleOnTable)
    {
        if (!dropBottle)
            LSLSender.SendLsl("Task 1 - Drop the Bottle", new float[] {
    142 });

        dropBottle = true;
    }
    if (bottleGrabbable.IsHeld && !bottleOnTable) dropBottle = false;
}
```

Listing 2.10: SettingsManager: Tasks Configurations

2.4.2 Task 2: Drinking Water

The second task simulates the familiar action of drinking water, designed to help improve grip control, arm movement, and hand-mouth coordination. This task involves repetitive actions, allowing the user to practice key motor functions multiple times throughout the session.

In this Task, a glass of water is placed on the table in a virtual highlighted area. The user starts by reaching for and grabbing the glass using their AR hands. This action requires precise grip control and hand coordination. After picking up the glass, the user performs a drinking motion, bringing it to their mouth to simulate drinking the water. This step encourages the development of fine motor skills and control over arm movements. After the drinking action is completed, the user returns the empty glass to its original position on the table. This reinforces the action of the controlled arm and hand movements, ensuring the glass is placed carefully on the table. After placing the glass back on the table, the user must once again place their hands in the designated area on the table, just as in previous tasks to begin the next try. Once the user's hands are correctly positioned on the table, the game automatically refills the glass with virtual water, preparing the user for the next attempt. The user will repeat this sequence of actions for the number of tries configured in the session settings.

Figure 2.6: Task2 - Drinking Water

In the Task2 class, the drinking action is controlled by checking if the glass reaches the height of the user's eyes. This ensures that the user lifts the glass correctly to simulate drinking before placing it back on the table. This class is responsible for managing the task where the user picks up a glass, drinks from it, and places it back on the table. The key action in this task is detecting when the glass is lifted to the height of the user's eyes and initiating the pouring of water when the user tilts the glass, signaling the start of the drinking action.

```
private void Update()
{
    UpdateEyeAndBottleLevel();
    UpdateEyeAndGlassLevel();

    ...

    if(bottleInRange && bottleAboveEyeLevel && bottle.Liquid.
   LiquidAbovePourOrigin)
    {
        drinkingForSeconds += Time.deltaTime;

        if (drinkingForSeconds >= secondsPerPoint)
```

```
        {
            //ScoreManager.Instance.IncreaseScore(task, 1);
            drinkingForSeconds = 0f;
        }
    }

    ...
}
```

Listing 2.11: SettingsManager: Tasks Configurations

2.4.3 Task 3: Moving a Box on a Three-Level Stage

The third task in the game involves moving a box between three levels of a stage. In this task, the user interacts with a box and must place it on different levels of a podium. This task is designed to improve the user's upper limb mobility, strength, and hand-eye coordination through controlled, repetitive lifting and placing actions. The task also challenges the user's ability to move objects across different heights, a crucial skill for functional recovery in daily life.

The user begins by reaching out to grab a box placed on the lowest level of a three-level stage. The box is light but requires a firm grip for stable movement. Once the task is started, the user is prompted to lift it and place it onto the stage's next level, which is distinguished by a highlighted area. The user must repeat this process, moving the box up one level at a time until it reaches the topmost level of the stage. After placing the box on the top level, the process repeats as the user returns the box back to the lowest level by reversing the movement. This requires carefully lifting the box from the higher level and placing it back down onto each lower level, one at a time, until it is on the lowest level again. The user repeats the action multiple times, as defined by the session settings. This repetition ensures the user practices controlled lifting and placement, encouraging both strength and precision in their movements. These opposing movements are critical for improving overall arm mobility and control. As with previous tasks, the user is required to place their hands back in the designated area on the table between each try. This ensures consistent positioning and resets the task flow.

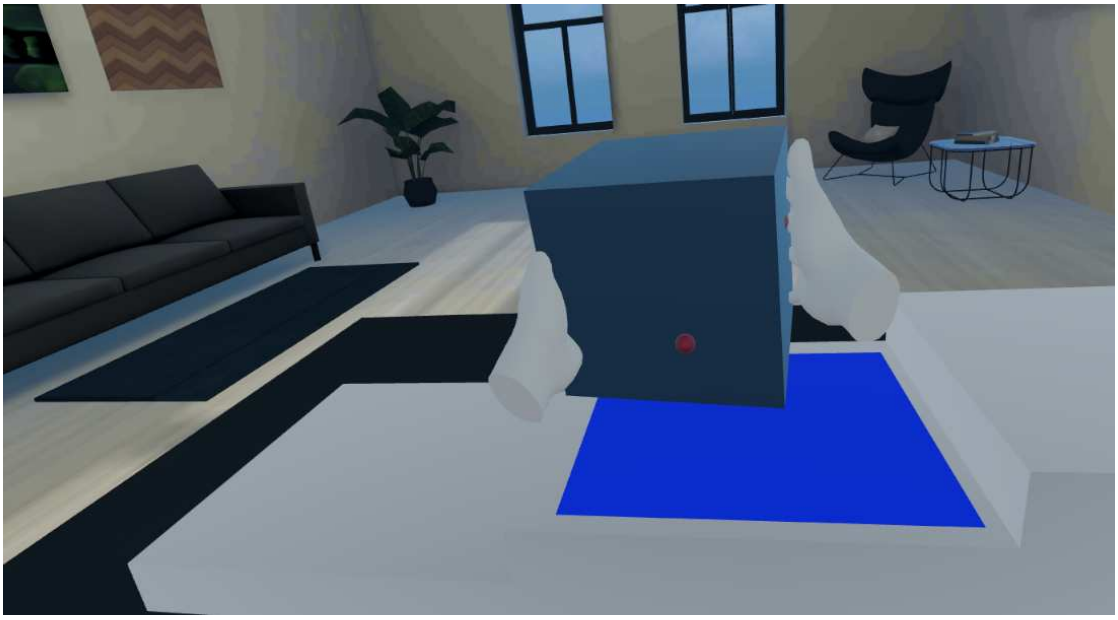

Figure 2.7: Task3 - Moving a Box on a Three-Level Stage

Task 3 has a dedicated Task3 class that manages the task's logic, with assistance from the PodestLevel class. The Task3 class handles the overall flow and progression of the task, while the PodestLevel class manages specific actions related to the multi-level platform or stages used in the task. The Task3 class coordinates these actions, ensuring that the box is correctly moved between the levels, while the PodestLevel class manages the positioning, detection, and validation of the box's placement on each level.

```
private void GrabbableEnteredArea(int level)
{
    if (movingUp && level == currentLevel + 1)
    {
        ScoreManager.Instance.IncreaseScore(task, 1);
        podestLevels[currentLevel].gameObject.SetActive(false);
        currentLevel = level;

        // check if reached top of podest and switch directions if so
        bool reachedTop = currentLevel == 2;//podestLevels[podestLevels.
    Length - 1].level;
        if (reachedTop)
        {
            movingUp = false;
        }

        DisplayNextStep();
        ActivatePodestLevels(false);
    }
```

```
    else if (!movingUp && level == currentLevel - 1)
    {
        ScoreManager.Instance.IncreaseScore(task, 1);
        podestLevels[currentLevel].gameObject.SetActive(false);
        currentLevel = level;

        bool reachedBottom = currentLevel == 0;
        if (reachedBottom)
        {
            movingUp = true;
        }

        DisplayNextStep();
        ActivatePodestLevels(false);
    }
}
```

Listing 2.12: SettingsManager: Tasks Configurations

2.5 Real-time Data Collection and Analysis

In this game, real-time data is collected during gameplay using the Lab Streaming Layer (LSL) protocol. This data includes crucial information such as event markers (e.g., task completions, user actions, errors, or transitions between game stages) and hand position tracking. Gathering data in real-time allows for immediate, precise monitoring of the user's interactions with the virtual environment, which is crucial for assessing their motor skills and rehabilitation progress.

The primary reason for collecting real-time data is to provide an accurate and dynamic representation of the user's performance and movements during the rehabilitation exercises. Real-time data allows for continuous monitoring of the user's performance during each task. By recording data like hand movements, task success rates, and reaction times, clinicians and researchers can analyze fine details about how the user interacts with the game, highlighting both progress and areas that need improvement. The system can detect when the user makes errors, such as incorrect object placement or inefficient hand movements. One of the critical purposes of collecting real-time game data is to synchronize it with external physiological signals, such as brain activity measured via EEG (Electroencephalography) or other sensors like EMG (Electromyography). Combining game data with brain signals correlates motor actions with brain activity to better understand how different regions of the brain are activated during specific motor tasks. Track cognitive load or engagement, identifying how challenging tasks are for the user and when they are exerting more mental effort. For example, when the user performs an action like pouring water or moving a box, the corresponding brain signals can be compared with the hand movement data captured in the game. This multi-source analysis can reveal how effectively the brain is controlling motor functions post-stroke and how recovery is progressing.

In the context of stroke rehabilitation, integrating game data with brain signals is vital for understanding the neuroplasticity process—the brain's ability to reorganize itself by forming new neural connections. By collecting and analyzing data from both the physical movements (captured in the game) and the brain's electrical activity (captured through EEG), the system enables several key insights. The real-time data allows researchers and therapists to observe how the brain responds to specific motor tasks. For instance, as the user attempts to move objects within the game, EEG signals may show how specific brain areas are activating or struggling to control the hand movements. This feedback can guide clinicians in tailoring therapy to the patient's specific needs. Over time, the collected real-time data can show patterns in the user's rehabilitation progress. As the user repeats tasks, improvements in reaction time, accuracy, or movement efficiency will be recorded and can be cross-referenced with corresponding brain signal improvements. This helps track how effectively the brain and body are working together during recovery. In addition to tracking motor function, real-time data can be used to assess fatigue—both mental and physical. As the user progresses through the tasks, certain markers (like slower movements or increased errors) can indicate when they are becoming fatigued.

Beyond its use in conjunction with brain signals, the real-time data collected in the game provides detailed metrics on each user's performance, such as speed, accuracy, and precision in each task. These metrics can be analyzed to provide personalized rehabilitation plans or to adjust the game difficulty in response to the user's abilities. Over multiple sessions, the collected data creates a log of the user's progress. Clinicians can compare current data with previous sessions to evaluate improvements in motor function and adjust therapy accordingly.

2.6 Data Collection and Event Markers

Throughout gameplay, various event markers are generated, which correspond to key in-game events and user actions. These markers provide insights into the user's performance, interactions, and movements. Markers capture some data like Game events (e.g., task start, task completion, object interactions, errors, or transitions between tasks). Hand position data also are sent to track the user's hand movements and interactions with virtual objects. This information is sent to the base station at predefined intervals, where it can be analyzed to evaluate the user's rehabilitation progress and the game's effectiveness. Each event marker is associated with a specific code (Index 0), making it easier to identify and categorize different types of events. The data are sent in data packets which start with an identifier code (Index 0) followed by optional data (Index 1-3), if required as shown in Table 2.6. Hand positioning data is collected for each hand separately every 100 milliseconds and sent to the base station. This ensures real-time monitoring of hand movements during the game, which is essential for accurate analysis and feedback in stroke rehabilitation tasks.

Event Markers heightDescription	Index 0	Index 1	Index 2	Index 3
Game Started	0			
Right Hand Position	80	x	y	z
Left Hand Position	90	x	y	z
Task 1 started	100			
Task 1 - Trial started*	101	1		
Task 1 - Trial finished**	102	2		
Task 1 finished	103	3		
Task 1 - Grabbing Glass	111			
Task 1 - Grabbing Bottle	112			
Task 1 - Released Glass	121			
Task 1 - Released Bottle	122			
Task 1 - Put on Area	130			
Task 1 - Drop the Glass	141			
Task 1 - Drop the Bottle	142			
Task 2 started	200			
Task 2 - Trial started*	201			
Task 2 - Trial finished**	202			
Task 2 finished	203			
Task 2 - Drinking Started	204			
Task 2 - Drinking Finished	205			
Task 2 - Grabbing Glass	213			
Task 2 - Released Glass	223			
Task 2 - Glass on the Area	230			
Task 2 - Drop the Glass	243			
Task 3 started	300			
Task 3 - Trial started*	301			
Task 3 - Trial finished**	302			
Task 3 finished	303			
Task 3 - Grabbing Box	314			
Task 3 - Released Box	324			
Task 3 - Box Put Down Successfully	330			
Object reset clicked (during trial)	400			

* Hands are on the table and the countdown is started
** object(s) are on highlighted area

2.7 Data Transmission Using the LSL Protocol

In this game, the Lab Streaming Layer (LSL) protocol is utilized to transmit real-time data from the game environment to a base station. The LSL protocol is specifically designed for synchronizing data streams between various devices and applications, making it highly suitable for use in rehabilitation games that require precise monitoring and data collection. The Lab Streaming Layer (LSL) is an open-source protocol that facilitates the real-time transfer of data streams across multiple platforms and devices. It was initially developed for applications in neuroscience and physiological monitoring but is now widely used in other areas requiring synchronized data exchange. LSL ensures that data streams, such as physiological data or game events, are transmitted in real-time with minimal delay. It supports different operating systems and applications, allowing for seamless integration in various rehabilitation setups. Timestamping and synchronization are some of the key features of this protocol. Each data stream is timestamped with high precision, ensuring accurate synchronization between game events and other systems. In the context of this game, LSL is employed to send important game data to a base station at specific intervals for analysis and monitoring. In the code the LSLSender class

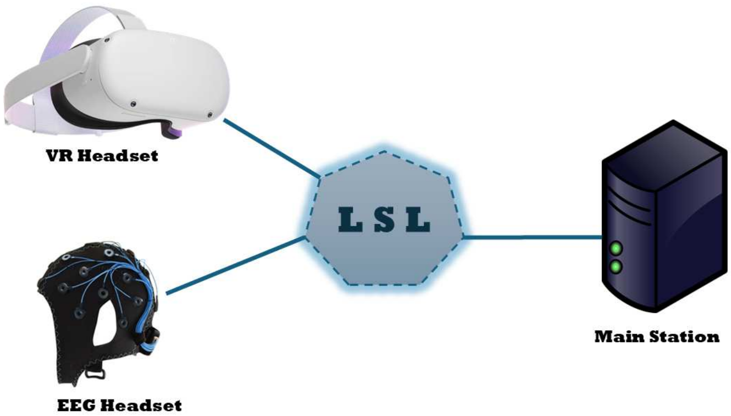

Figure 2.8: LSL Protocol Diagram

In this project, we defined an Agent or Wrapper Class for the LSL (Lab Streaming Layer) library, which is responsible for packaging and sending data. This class abstracts the complexity of the LSL protocol, making it easier to manage and transmit different types of data, such as hand positioning and event markers, to the base station.

```
public static StreamInfo info = new StreamInfo("VR_GAME", "EEG", 1, 100,
    LSL.channel_format_t.cf_float32, "VR");
public static StreamOutlet outlet;

public static void init()
{
    outlet = new StreamOutlet(info);
}
```

```

public static async void SendLsl(string title, float[] data)
{
    outlet.push_sample(data);
    Debug.LogWarning("VR LSL sent: " + data[0].ToString() + "  (" + title
     + ")");
}
```

Listing 2.13: LSLSender

3 Results

The event marker diagram provides a comprehensive overview of the temporal distribution of markers across the three tasks, allowing for a quantitative assessment of user interaction and task execution. Each task demonstrates a distinct pattern of marker density and timing, reflecting complexity and user engagement differences. Further statistical analysis, such as measuring average task completion times and marker intervals, could quantify these trends more precisely.

Task 1, which involves grabbing and positioning a bottle and glass, exhibits the highest marker density and the longest completion time. It reveals significant insights into user coordination patterns during the task and makes the user use both hands. The User has to move both hands simultaneously and sequentially to complete the task. Task 1 required precise hand positioning and interaction with two objects. The time intervals between these markers highlight task complexity, with average intervals between grabbing and releasing actions providing insights into user efficiency and task difficulty. These observations suggest that simultaneous hand movements prioritize speed but demand greater coordination to prevent mistakes, whereas sequential actions favor precision over speed. Analyzing these coordination patterns provides a deeper understanding of user behavior and highlights potential areas for improving task design and user guidance. Task 2 introduces a simpler interaction, focusing on lifting a glass to eye level and simulating a drinking action. The event markers here are less dense compared to Task 1. The relatively short intervals between these markers suggest a more streamlined process, with users completing the task in less time compared to Task 1. Task 3, involving the manipulation of a box, shows the sparsest distribution of event markers. This reduced complexity is reflected in smaller time gaps between markers and a shorter overall task duration.

3.1 Analysis of Hand Positioning Data

Figure 3.2 displays the x, y, and z coordinates of both hands recorded over time during the execution of three sequential tasks. After each try, both hands must touch the highlighted area on the table, resulting in a clear diagram for each trial. The x-axis represents forward and backward hand movements, the z-axis lateral movements, and the y-axis vertical lifting movements.

In Task 1, the hands move laterally to reach for the bottle and glass. In the first half of the diagram, the x-axis is represented by blue and orange lines. This forward hand movement is expected to be moderate as the hands mainly reach toward the objects and

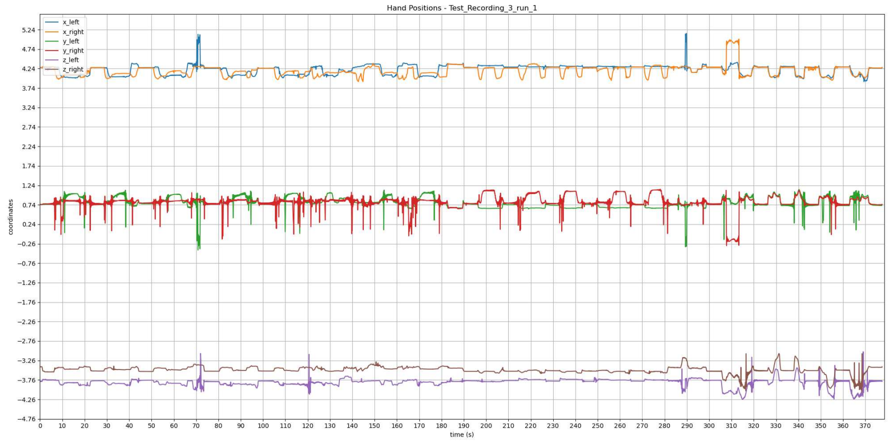

Figure 3.1: Hand Positioning Diagram

return to the starting position. The red and green y-axis shows vertical hand movements, which would be more pronounced in this task since the user lifts the bottle, pours water into the glass, and then returns the items to the table. Notable spikes in this axis should correspond to the action of lifting and lowering. The z-axis is shown in brown and purple lines. Small fluctuations in the z-axis lines likely indicate adjustments made while grasping and positioning the bottle and glass. Minimal movement along the z-axis suggests limited lateral changes, which aligns with the task focusing on reaching and lifting rather than lateral movements.

Drinking water from a glass in Task 2 starts at the beginning of the second half of the diagram. The forward movement here is likely minor, as the task involves bringing the glass up from the table to the user's face, rather than reaching across a wide area. In this task, the left hand is inactive, showing minimal changes across all axes. The y-axis changes should be significant, reflecting the action of lifting the glass to the height of the user's face and lowering it again after drinking. This upward motion can be seen in recurring peaks or consistent elevation patterns of the red line. For the z-axis, minor changes may indicate subtle adjustments in the lateral as the user brings the glass toward their mouth, especially if a slight forward motion is required.

Finally, Task 3 involves the lateral movement of the box between podium levels. This task should show a more consistent pattern of lateral adjustments compared to the previous tasks because both hands hold the box and move together. The vertical movement here would correlate with lifting the box to different podium levels and lowering it. Peaks in the y-axis may be higher and more regular compared to Tasks 1 and 2 due to the repeated upward and downward placement. The x-axis shows a repetitive pattern with two fixed values in Task 3: one when holding the box and another when placing the hands on the table after a successful attempt.

Patterns within each axis provide insights into the consistency of the user's movements. Consistent peaks and troughs in specific coordinates can indicate stable, repetitive actions, which are desirable in rehabilitation as they suggest improving motor control. Each axis's stability and range reflect the user's ability to maintain control over their hand movements. For instance, stability in the x and z-axes combined with controlled y-axis movements shows the user's ability to accurately lift and place objects. The data's real-time nature allows assessment of how well the user is performing each task. For example, in Task 1, smooth transitions in the y-axis during pouring could indicate controlled hand movements, while erratic movements might suggest difficulty. By examining patterns in the data for each task, therapists can analyze the user's movement precision, control, and consistency, essential for rehabilitation. The ability to compare these data across multiple sessions can help in tracking progress, identifying strengths and weaknesses in motor control, and adjusting the rehabilitation program as needed.

3.2 Analysis of Event Markers Data

The event markers diagram illustrates the timeline of markers sent to the base computer during the three tasks. Each event marker is associated with a unique code, where the first digit identifies the relevant task. This diagram provides a structured overview of the sequence and timing of actions for each task, with the Table 2.6 descriptions helping to interpret each marker. The table in Section 2.6 provides the complete descriptions of each code. Task transitions become visible in the diagram when the codes shift from one task series to another. For instance, the transition from Task 1 to Task 2 can be seen when markers switch from 1xx to 2xx codes. Each task begins with a Trial Started marker (x01: 101 for Task 1, 201 for Task 2, and 301 for Task 3) and ends with a Trial Finished marker (x02: 102, 202, and 302). These markers define the boundaries of each task trial, indicating when each new trial starts and ends. By analyzing these patterns, it is possible to assess user behavior, task difficulty, and successful completion of actions in each task scenario.

Within each task, we observe clusters of markers indicating sequences of actions, such as grabbing, releasing, and positioning objects. Markers like 111 (Grabbing Glass), 112 (Grabbing Bottle), and 121/122 (Releasing Glass/Bottle) appear in clusters, showing moments of intensive interaction with objects in Task 1. Clusters around 213 (Grabbing Glass) and 223 (Releasing Glass) relate to handling the glass during the drinking task. Markers like 314 (Grabbing Box) and 324 (Released Box) capture actions associated with picking up and releasing the box. Markers such as 130/230 (Put on Area), 141/243 (Drop Glass), and 330 (Box Put Down Successfully) represent object placements, which are critical to task completion.

The initial marker (code 100) signifies the beginning of the game session. Code 400, if

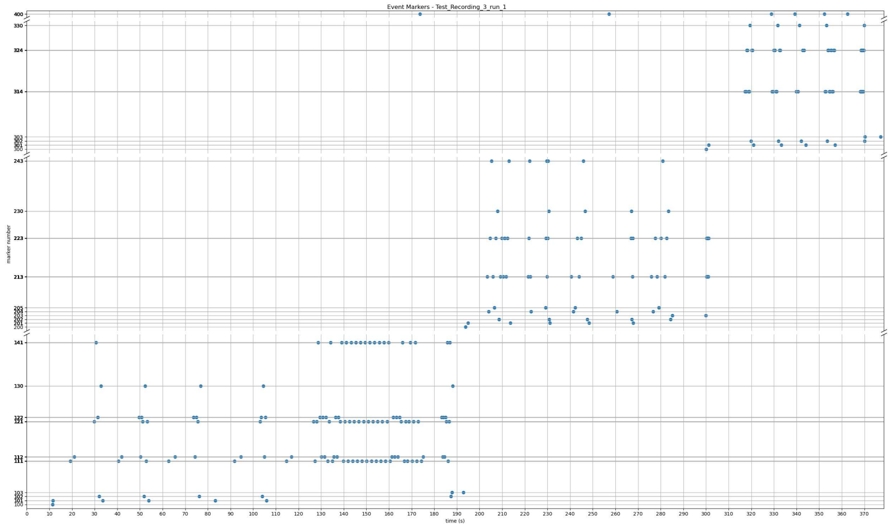

Figure 3.2: Event Markers Diagram

present, indicates that an object was reset during a task, which may occur due to user error or the need to restart the task. Each task ends with a completion marker (x03: 103 Task 1 finished, 203 Task 2 finished, and 303 Task 3 finished), signaling that the task is complete. Time intervals between the end of one task and the beginning of the next reveal the transition time, showing potential pauses or delays between tasks. The density of markers within each task may reflect its complexity. Task 1, which involves two objects, shows more varied interactions, while Task 3, focused on box placement, contains fewer but more deliberate actions.

4 Discussion

In this thesis, we designed and developed the Rehab App, presenting a novel approach to stroke rehabilitation by integrating immersive VR with interactive and customizable rehabilitation tasks. The game targets hand motor function recovery through a combination of engaging, repetitive exercises and real-time performance tracking. The modular design and use of Oculus SDK in Unity have facilitated the creation of a flexible and scalable platform, while the inclusion of AR representations of the user's hands ensures a natural and effective interaction experience. The ability to customize settings, such as task randomization and left-handed compatibility, highlights the system's adaptability to individual patient needs.

From the analysis of task-specific hand dynamics, we observe distinct patterns across various activities. In Task 1, reaching and lifting motions dominate, characterized by moderate forward movement along the x-axis and pronounced vertical spikes along the y-axis. These reflect the sequential actions of lifting a bottle, pouring water, and returning items to the table. Minimal lateral changes along the z-axis indicate stability during grasping and positioning. Task 2 primarily involves actions performed by the right hand, with significant vertical changes in the y-axis linked to lifting a glass to the user's face and lowering it back. Forward movement along the x-axis remains minimal, while subtle lateral adjustments along the z-axis demonstrate efficient and focused hand use. Task 3 features coordinated lateral movements along the z-axis by both hands to transfer a box between podiums. Repeated upward and downward peaks in the y-axis confirm consistent lifting and placement, while fixed x-axis values reflect structured hand positioning at defined points.

We find out that the analysis of event markers presents distinct patterns in user behavior and task execution. Each task is defined by "Trial Started" and "Trial Finished" markers, with transitions marked by code changes. Clusters of markers within tasks represent specific actions, such as grabbing, releasing, and placing objects. Task 1 shows a higher density of markers, reflecting complex interactions, while Task 3 involves fewer but deliberate actions. Transition times between tasks reveal pauses or delays, offering insights into task difficulty or user behavior. This marker system provides a structured framework for evaluating motor control and rehabilitation progress.

We also observe consistency and control through stable and repetitive movement patterns across tasks, marked by consistent peaks and troughs. Smooth transitions in the y-axis, particularly during lifting tasks, further highlight controlled and deliberate hand movements.

From a development perspective, we find that the immersive VR environment, complemented by AR hand projections, creates a realistic and motivating setting for rehabilitation. The integration of a point-based system provides immediate feedback, encouraging patients to achieve better outcomes. In addition, features such as task randomization, physical alignment calibration, and left-hand compatibility make the game adaptable to a diverse range of users with varying rehabilitation needs. Detailed performance metrics collected during the game allow healthcare professionals to monitor progress and refine therapy plans effectively. Metrics such as task completion time, movement precision, and error rates provide actionable insights into patient recovery trajectories. Additionally, the intuitive menu systems and options, designed specifically for stroke survivors with potential cognitive or mobility challenges, enhance the game's overall accessibility and usability.

The Rehab App also has limitations from a development perspective that require further study. Interaction is limited to the table area, limiting real-world movement replication, and expanding the interaction range could improve functionality. The app tracks short-term metrics but lacks validation of long-term impacts on recovery and quality of life, which longitudinal studies could address. In addition, reliance on VR hardware limits accessibility; exploring cost-effective or widely available alternatives could increase adoption.

We also acknowledge that the tasks primarily involve reaching, lifting, and minor lateral adjustments, which may not adequately represent the complexity of real-world hand dynamics. This limited scope reduces the applicability of findings to broader rehabilitation scenarios. Additionally, the role of the non-dominant hand is restricted, particularly in Task 2, where the left hand plays a minimal role. This provides limited insights into its motor control and coordination, suggesting that future experiments should include tasks requiring bilateral engagement. Moreover, our setup assumes consistent environmental factors, such as table height and object placement, which does not account for variability in real-world rehabilitation settings. This could influence movement trajectories and reduce the ecological validity of the findings. Furthermore, while the data provide valuable insights, the interpretation of subtle fluctuations in the z-axis and minor lateral adjustments remains somewhat subjective. Incorporating additional sensors or markers could enhance precision and provide a more granular analysis.

In the future, systems could leverage advancements in VR hardware, such as lighter, more ergonomic headsets with improved battery life and higher resolution. These updates could enhance user comfort and immersion, making the system more suitable for extended rehabilitation sessions. Incorporating real-world objects like an actual glass and bottle into the VR environment, through techniques like augmented reality (AR), can improve task realism and user engagement. This approach would also eliminate discrepancies between virtual and physical interactions, providing a more seamless training experience. Additional tasks could be developed to cover a broader range of movements, including those that simulate daily life activities or involve complex motor coordination. These tasks could further challenge and train the patient's upper-limb motor functions. The system could be expanded to include tasks aimed at lower-limb rehabilitation. These might involve walking simulations, stepping tasks, or exercises to strengthen balance and coordination.

Future designs could integrate AI to dynamically adjust task difficulty based on patient performance, ensuring that users remain challenged while avoiding frustration or disengagement. Introducing multiplayer or collaborative tasks could enhance motivation by allowing patients to interact with others in a virtual space. This could simulate real-life social and collaborative scenarios, providing additional therapeutic value.

More sophisticated data analytics could be developed to provide deeper insights into patient performance and progress. This might include 3D movement tracking or comparisons with normative data, enabling therapists to tailor interventions more effectively. Refining the system for use in a home setting could make rehabilitation more accessible. Compact, user-friendly versions of the system, combined with remote therapist monitoring capabilities, would allow patients to continue therapy outside clinical environments.

5 Conclusion

This thesis presents the development and evaluation of a VR-based rehabilitation system designed to enhance upper-limb motor recovery for stroke survivors. The project integrates innovative technologies, including a VR headset and the Lab Streaming Layer (LSL) protocol, to create an immersive environment for therapeutic tasks. By combining real-time feedback, task randomization, and precise performance tracking, the system aims to offer a novel and effective approach to post-stroke rehabilitation.

The system was carefully designed to replicate realistic interactions, with tasks such as pouring water, drinking, and moving objects across different levels. These activities were chosen to mimic daily life challenges while focusing on improving motor coordination, strength, and dexterity. The inclusion of calibration processes ensured alignment between the virtual and physical worlds, enabling accurate and meaningful training experiences. Additionally, the system addressed key challenges, such as object alignment, hand-object interaction, and task separation, through thoughtful design choices like hand placement markers and reset mechanisms.

Looking back at the research questions, the answers can be summarized as follows:
RQ1- *What are concrete tasks designed in the VR game to target different aspects of motor recovery in stroke rehabilitation, such as coordination, strength, and fine motor skills?*
This research question is addressed by designing three concrete tasks within the VR game, each targeting distinct aspects of motor recovery in stroke rehabilitation. Task 1 emphasizes coordination and fine motor skills through reaching, grasping, and precise placement actions. Task 2 focuses on strength and lifting accuracy, requiring controlled vertical movements. Task 3 promotes bilateral coordination and spatial awareness by engaging both hands in lateral object transfers.
RQ2- *How to design a scoring system based on accuracy and speed serving as a motivational tool for stroke patients during their rehabilitation?*
To motivate patients, we implement a scoring system based on speed and accuracy, providing immediate feedback and fostering engagement through clear performance metrics.
RQ3- *How to collect real data time in a VR game and transmit it to a central station?*
Real-time data collection is achieved by embedding event markers that track user actions and task progression, transmitting this data to a central station for detailed analysis. This approach can later enable therapists to monitor patient performance and tailor rehabilitation plans effectively.

This work successfully achieved its defined goals through a combination of innovative design, user-focused development, and data integration. For the **development of a VR rehabilitation game**, we used Unity and the Oculus SDK to create an engaging and interactive platform tailored to improve motor skills and coordination for stroke survivors. For **user-centric design and interaction**, we designed a virtual environment resembling a home setting to promote familiarity and reduce anxiety. Hand interactions were augmented using virtual hand representations, offering an intuitive interface that promotes comfort and encourages active participation. **Task-oriented rehabilitation** was implemented through the design of a series of tasks that target key motor functions. These tasks incorporated repetition and real-time feedback mechanisms to maintain engagement and enhance motor recovery. We integrated **real-time data collection and analysis** using the LSL protocol, enabling us to monitor user interactions dynamically. This system captured detailed performance metrics and correlated them with external physiological signals, providing insights into neuroplasticity and motor skill improvement. Together, these components reflect a holistic approach to stroke rehabilitation, combining advanced VR technology with evidence-based therapeutic principles.

Quantitative results from the system provide insights into user performance, including hand positioning, task success rates, and coordination patterns. These metrics are valuable not only for tracking rehabilitation progress but also for refining therapeutic strategies. The system's ability to send and synchronize data with external devices, such as EEG headsets, enhances its potential for multi-modal analysis, paving the way for a deeper understanding of neuro-motor recovery processes.

Despite its achievements, the system also faced limitations. Challenges such as the weight of the VR headset, the complexity of calibrating physical and virtual elements, and maintaining reliable wireless connectivity required significant effort to overcome. Furthermore, certain usability and performance aspects, such as object behavior and task complexity, need further refinement to maximize the system's therapeutic impact.

Looking ahead, this system has immense potential for further development. Future iterations could incorporate advancements in VR hardware, explore augmented reality integration with real-world objects, and expand task libraries to cover a wider range of movements, including lower-limb rehabilitation. Adaptive difficulty levels, enhanced data analytics, and home-based implementations would further increase accessibility and efficacy, empowering patients to continue their recovery journey independently.

In conclusion, this project demonstrates the feasibility and promise of using immersive VR for stroke rehabilitation. By addressing real-world therapeutic needs and integrating state-of-the-art technologies, the system contributes a meaningful step toward improving the quality of life for stroke survivors. While there is room for improvement, the framework established in this work lays a solid foundation for future innovations in the field of digital rehabilitation.

Bibliography

[1] Leila Shahmoradi et al. "Virtual reality games for rehabilitation of upper extremities in stroke patients". In: *Journal of Bodywork and Movement Therapies* 26 (2021), pp. 113–122. ISSN: 1360-8592. DOI: https://doi.org/10.1016/j.jbmt.2020.10.006. URL: https://www.sciencedirect.com/science/article/pii/S136085922030187X.

[2] Lucija Mihic Zidar et al. "Revire: A virtual realitiy platform for BCI-based motor rehabilitation". English. In: 9th Graz Brain-Computer Interface Conference 2024 ; Conference date: 09-09-2024 Through 12-09-2024. Verlag der Technischen Universität Graz, Sept. 2024. DOI: 10.3217/978-3-99161-014-4-048. URL: https://www.tugraz.at/institute/ine/graz-bci-conferences/9th-graz-bci-conference-2024.

[3] Johnathan Bown and Jayne Gackenbach. *Boundaries of Self and Reality Online: Implications of Digitally Constructed Realities*. Academic Press, 2017.

[4] Steven M LaValle. *Virtual reality*. Cambridge university press, 2023.

[5] Edward Probir Mondol. "The role of Vr games to minimize the obesity of video gamers". In: *International Journal of Computations, Information and Manufacturing (IJCIM)* 2.1 (2022).

[6] Gordon Tao et al. "Immersive virtual reality health games: a narrative review of game design". In: *Journal of NeuroEngineering and Rehabilitation* 18 (2021), pp. 1–21.

[7] Biao Xie et al. "A review on virtual reality skill training applications". In: *Frontiers in Virtual Reality* 2 (2021), p. 645153.

[8] Younhyun Jung. "Virtual reality simulation for disaster preparedness training in hospitals: integrated review". In: *Journal of Medical Internet Research* 24.1 (2022), e30600.

[9] Ravikiran Mane, Tushar Chouhan and Cuntai Guan. "BCI for stroke rehabilitation: motor and beyond". In: *Journal of Neural Engineering* 17.4 (2020), p. 041001.

[10] Catherine Owens Johnson et al. "Global, regional, and national burden of stroke, 1990–2016: a systematic analysis for the Global Burden of Disease Study 2016". In: *The Lancet Neurology* 18.5 (2019), pp. 439–458.

[11] Christine Therese Shiner. "Characterising post-stroke heterogeneity from brain to muscle: Implications for clinical assessment andmotor rehabilitation". In: *Ph.D. dissertation* (2015).

[12] Robert Gabriel Lupu et al. "BCI and FES based therapy for stroke rehabilitation using VR facilities". In: *Wireless Communications and Mobile Computing* 2018 (2018).

[13] Athanasios Vourvopoulos et al. "Effects of a brain-computer interface with virtual reality (VR) neurofeedback: A pilot study in chronic stroke patients". In: *Frontiers in human neuroscience* (2019), p. 210.

[14] Cosmina Isar. "A glance into virtual reality development using unity". In: *Informatica Economica* 22.3 (2018), pp. 14–22.

[15] R. van der Vliet et al. "Predicting Upper Limb Motor Impairment Recovery after Stroke: A Mixture Model". In: *Annals of Neurology* 87.3 (Mar. 2020). Epub 2020 Jan 25. PMID: 31925838; PMCID: PMC7065018, pp. 383–393. DOI: `10.1002/ana.25679`.

[16] Catherine E. Lang, Keith R. Lohse and Rebecca L. Birkenmeier. "Dose and timing in neurorehabilitation: prescribing motor therapy after stroke". In: *Current Opinion in Neurology* 28.6 (Dec. 2015), pp. 549–555. DOI: `10.1097/WCO.0000000000000256`.

[17] T. Massetti et al. "The Clinical Utility of Virtual Reality in Neurorehabilitation: A Systematic Review". In: *Journal of Central Nervous System Diseases* 10 (Nov. 2018). PMID: 30515028; PMCID: PMC6262495, p. 1179573518813541. DOI: `10.1177/1179573518813541`.

[18] H. Mousavi Hondori and M. Khademi. "A Review on Technical and Clinical Impact of Microsoft Kinect on Physical Therapy and Rehabilitation". In: *Journal of Medical Engineering* 2014 (2014), p. 846514. DOI: `10.1155/2014/846514`. eprint: `https://doi.org/10.1155/2014/846514`.

[19] M.S. Cameirão, S. Bermúdez and P. Verschure. "Virtual Reality Based Upper Extremity Rehabilitation Following Stroke: A Review". In: *Journal of CyberTherapy & Rehabilitation* 1.1 (2008), pp. 63–74.

[20] Gustavo Saposnik, Mindy Levin and for the Stroke Outcome Research Canada (SORCan) Working Group. "Virtual Reality in Stroke Rehabilitation". In: *Stroke* 42.5 (2011), pp. 1380–1386. DOI: `10.1161/STROKEAHA.110.605451`. eprint: `https://www.ahajournals.org/doi/pdf/10.1161/STROKEAHA.110.605451`. URL: `https://www.ahajournals.org/doi/abs/10.1161/STROKEAHA.110.605451`.

[21] Mateus Trombetta et al. "Motion Rehab AVE 3D: A VR-based exergame for post-stroke rehabilitation". In: *Computer methods and programs in biomedicine* 151 (2017), pp. 15–20.

[22] Lynne M Weber et al. "Immersive virtual reality mirror therapy for upper limb recovery after stroke: a pilot study". In: *American journal of physical medicine & rehabilitation* 98.9 (2019), pp. 783–788.

[23] Destaw B Mekbib et al. "Proactive motor functional recovery following immersive virtual reality–based limb mirroring therapy in patients with subacute stroke". In: *Neurotherapeutics* 17.4 (2020), pp. 1919–1930.

[24] Julian Felipe Villada Castillo et al. "Design of virtual reality exergames for upper limb stroke rehabilitation following iterative design methods: usability study". In: *JMIR Serious Games* 12.1 (2024), e48900.