Hiragana Sensei

By

Lai Kah Hoe

FACULTY OF COMPUTING AND
INFORMATION TECHNOLOGY

TUNKU ABDUL RAHMAN UNIVERSITY OF
MANAGEMENT AND TECHNOLOGY
KUALA LUMPUR

ACADEMIC YEAR
Department of Computer Science and Embedded System.

Hiragana Sensei

By

Lai Kah Hoe

Supervisor: FOONG THENG CHAI

A project report submitted to the
Faculty of Computing and Information Technology
in partial fulfillment of the requirement for the
Bachelor of Computer Science (Honors)
in Interactive Software technology

Faculty of Computing and Information Technology
Tunku Abdul Rahman University of Management and Technology
Kuala Lumpur

Declaration

The project submitted herewith is a result of my own efforts in totality and in every aspect of the project works. All information that has been obtained from other sources had been fully acknowledged. I understand that any plagiarism, cheating or collusion or any sorts constitutes a breach of TAR University rules and regulations and would be subjected to disciplinary actions.

kahhoe

Lai Kah Hoe

ID: 23WMR00157

Abstract

The purpose of this project is to develop an educational VR game designed to enhance the learning of Japanese Hiragana characters, focusing on both identification and writing skills. Many existing Japanese educational games suffer from limited character sets and lack of randomised sequences, which hinder players' ability to recognize and adapt to various characters. Additionally, some games fail to provide writing tests, resulting in players being able to read but not write Hiragana. This project addresses these issues by generating more randomised character sequences and incorporating writing assessments to improve learners' adaptability and proficiency in both reading and writing Hiragana.

The project involves the creation of a VR educational game that includes interactive features to teach Japanese Hiragana. It covers the development of a system with randomised character sequences and writing exercises. The scope includes designing and implementing various modules such as character presentation, writing assessment, and user interaction within the VR environment. The project aims to provide a comprehensive learning tool that addresses existing gaps in Japanese language education.

The methodology includes utilising VR technology to create an immersive learning environment. Tools and techniques used in the project involve dynamic character presentation, interactive gameplay, and real-time feedback mechanisms for writing assessments. The development process includes designing the VR interface, implementing character sequence algorithms, and integrating writing tests to evaluate learner proficiency.

Testing criteria for the project include assessing user engagement, learning outcomes, and the accuracy of character writing. Various evaluation methods are employed to measure the effectiveness of the game in improving recognition and writing skills.

The project results indicate that the VR game improves players' ability to recognize and write Hiragana characters. The strengths of the system include its ability to provide a more comprehensive learning experience through randomised sequences and writing tests. Limitations, such as potential technological constraints and user adaptation to VR controls, are noted. Overall, the project contributes to the educational field by offering an innovative solution for learning Japanese Hiragana, combining interactive gameplay with practical writing practice.

Acknowledgement

This contains acknowledgement to those who have contributed directly or indirectly to the completion of the project. Usually the people to be acknowledged include the project supervisor(s), moderators, family, and those who have given assistance and support to ensure the success of the project.

Table of Contents

Chapter 1

Introduction

1 Introduction

This chapter provides an overview of the "Hiragana Sensei" project, detailing its general background, project description, features, advantages, project planning, and project team. "Hiragana Sensei" is a Virtual Reality (VR) game designed to teach players Japanese Hiragana characters by placing them in the role of a teacher in a virtual classroom. The game offers a unique and interactive learning experience, utilising VR technology to enhance engagement, motivation, and retention. This section will explore the key aspects of the project, including its innovative features, educational benefits, strategic planning, and the dedicated team responsible for its development.

1.1 Project Objective

The primary objective of the "Hiragana Sensei" project is to develop an engaging and immersive VR game that enables players to learn writing of Japanese Hiragana characters as well as serving as a platform to practice writing. The game aims to combine effective educational techniques with entertaining gameplay, providing an interactive experience that ensures players not only recognize but also accurately write Hiragana characters.

1.2 Project Background

“Hiragana Sensei” is a VR platform where players play the role of a Japanese teacher in a classroom. Students will raise their hands and ask in Romaji (English character), while the teacher needs to write the Hiragana (Japanese character) corresponding to the Romaji to answer the student using the VR controller. Points will be awarded to the teacher if correct and deducted if wrong. While the student is asking questions, 5 bubbles will pop up around the student to show hints for the player to indicate the correct Hiragana character. For new show up characters, the bubble will be in different colours for the player to learn the character. As the player gets more correct answers, the characters in the bubble will not show a hint color and finally no hint bubbles will be displayed. The level ends when the player reaches the threshold marks for every character in the level. The game goes on until all levels are cleared. The game acts as a repeatable training platform for players to practice writing Hiragana characters.

1.3 Features List

Student-Teacher Interaction:

Students in the virtual classroom raise their hands and ask for the Hiragana of a given Romaji. Player answers by writing the correct answer on the whiteboard.

Hint System:

Upon questions asked by students, bubbles containing hints pop up around the student. The bubbles provide Hiragana characters to help the player write the correct Hiragana character. For new characters, hint bubbles appear in different colours to aid learning. As the player's difficulty increases, the hint bubbles will no longer be available..

Writing and Drawing:

Players use VR controllers to write or draw the correct Hiragana character, promoting active learning and retention.

Input Recognition:

Optical Character Recognition (OCR) will be used to recognize the input that is written by the player.

Performance Display:

At the end of each level, the percentage of correctness will be displayed. If the player finished all levels, a final display of each character's correct performance will be displayed.

1.4 Advantage and Contribution

1.4.1 Advantages

Immersive Learning Environment:

VR provides an engaging and interactive learning environment, making the learning process more enjoyable and effective. The game-like elements, such as scoring, difficulty, progression as well as final performance checking motivate players to continue learning and improving their skills in writing Hiragana.

Interactive Learning:

Players actively participate in the learning process by writing characters and answering questions from virtual students, which helps in better retention of what was learned. The game provides immediate feedback on the player's performance, allowing for quick correction and reinforcement of learning.

Personalized Learning Pace:

The replayable game system allows players to learn at their own pace, accommodating different learning speeds at any time. The gradual reduction of hints ensures that the difficulty adapts to the player's improving skill level, providing an appropriate challenge at all times.

Different From Competitors:

“Hiragana Sensei” occupies a unique position in the educational game industry. While there are numerous mobile games that focus on writing and selecting tasks, creating a mobile game for this project would offer limited value. However, on the VR platform, there are very few, if any, competitors for “Hiragana Sensei”. Most existing VR educational games focus on speaking and selecting activities. Furthermore, writing mechanisms are still underdeveloped in VR, as the industry has primarily concentrated on drawing mechanics instead. Therefore “Hiragana Sensei” will bring far greater value than expected in the VR education industry.

High Customization Potential:

This project offers significant customization potential due to its simple language-switching code model. By utilizing a set of trained data and mapping characters of other languages to their corresponding Romaji, the project can be fully adapted to support a different language base.

1.4.2 Contributions

Innovative Educational Tool:

"Hiragana Sensei" introduces the potential of VR as an effective tool for teaching the fundamental basics of language by providing a platform to learn to recognize and write characters of others language.

Language Learning Enhancement:

Acts as an effective supplementary resource for traditional Hiragana learning methods, providing an alternative way to practise and reinforce knowledge. Promotes interest in Japanese language and culture, encouraging more people to learn Japanese.

Technological Advancements:

Showcases the integration of VR technology in educational applications, highlighting its potential to transform traditional learning methods. Encourages collaboration between educators, game developers, and VR specialists to create innovative educational tools.

Research and Development:

Provides valuable data on the effectiveness of VR in language learning, contributing to research in educational technology. Collects user feedback to improve future iterations of the game and develop more effective educational VR experiences.

Educational Values

"Hiragana Sensei" significantly enhances educational values by transforming traditional language learning into an interactive VR experience. The game boosts engagement and motivation through game-based elements like scoring and progressive challenges, encouraging active learning with immediate feedback. It aids memory retention and critical thinking by offering regular practice and adaptive difficulty. The use of VR controllers for writing characters improves fine motor skills and hand-eye coordination. Additionally, "Hiragana Sensei" immerses students in the Japanese language, fostering cultural awareness and proficiency. As a flexible and enjoyable tool, it promotes lifelong learning and positive attitudes towards education, while demonstrating the potential of VR in innovative teaching methods and educational technology research.

1.5 Project Plan

The project plan is outlined on a weekly basis, beginning from July 1st.

Week	**Activities**
	Semester 1
1 - 2	General project concept development, outlining research methods, and identifying online resources for the project.
3 - 4	Writing Chapter1 Introduction for the documentation of project 1.
5 - 6	Writing Chapter2 Literature Review for the documentation of project 1.
7 - 8	Writing Chapter3 Methodology and Requirement for the documentation of project 1.
10 - 12	Writing Chapter4 System Design for the documentation of project 1.
13	Perform Deeper Research on Optical Character Recognition (OCR) to read Hiragana.
14	Create the game prototype.
	Semester Break
1	Start creating full game features.
2 - 3	Complete the features of the game.
	Semester 2
1 - 2	Perform game polishing.
3 - 4	Perform final game testing and prepare for the final presentation.
5	Writing Chapter 5, 6 and 7 for the documentation of project 2.
6	Prepare for submission.

1.6 Project Team and Organization

This project is a solo initiative undertaken by a student, Lai Kah Hoe as part of the final year academic requirements. The development process is conducted under the mentorship and supervision of a faculty member from TAR UMT, Ms Chai Foong Theng. This guidance is instrumental in ensuring that the project reaches the academic standards and achieves the required quality. The involvement of the supervisor from TAR UMT provides valuable oversight, contributing to the project's overall success and alignment with institutional expectations.

1.7 Chapter Summary and Evaluation

1.7.1 Summary

The "Hiragana Sensei" project is a Virtual Reality (VR) game designed to teach Japanese Hiragana characters through an immersive and interactive learning environment. The game places players in the role of a teacher in a virtual classroom where they must respond to students asking for Hiragana characters by writing them using VR controllers. Key features include an interactive hint system, writing and drawing exercises, and the use of Optical Character Recognition (OCR) for input recognition.

The project addresses existing challenges in Japanese educational games, such as limited character sets and the lack of writing practice. By offering randomised character sequences and incorporating writing tests, the game aims to improve players' recognition and writing skills. The advantages of the game include its engaging VR environment, personalised learning pace, and high customization potential. It also contributes to educational technology by showcasing VR's potential as an effective teaching tool and providing valuable insights into its use of writing/drawing actions in language learning .

The project plan is organised on a weekly basis, starting from July 1st, and includes phases such as research, documentation, prototype development, game polishing , presentation and final submission. The project is undertaken by Lai Kah Hoe, with supervision from Ms. Chai Foong Theng from TAR UMT, ensuring adherence to academic standards and quality.

1.7.2 Evaluation

This chapter highlights the strengths and potential of "Hiragana Sensei," emphasizing its unique contribution to the educational game industry. It also explores the game's potential impact and significance within the industry, showcasing its innovative position and future influence. "Hiragana Sensei" stands out as a pioneering educational tool in the VR space, offering a unique approach to learning Japanese Hiragana characters.The game’s ability to address both reading and writing skills distinguishes it from other educational VR games, which often focus solely on recognition or selection. Furthermore, its adaptability through simple language switching enhances its scalability and customization, making it applicable to a wider range of learners. As the VR educational game market continues to evolve, "Hiragana Sensei" has the potential to lead the way in developing writing-based mechanisms in VR, filling a gap in the current educational game landscape. This project not only contributes to the field of language learning but also sets a benchmark for future VR educational applications.

Chapter 2

Literature Review

2 Literature Review

This literature review is based on existing research relevant to this project, offering a comprehensive evaluation of prior studies and findings that relate to the development of "Hiragana Sensei." By analysing these research results, the review provides valuable insights and context that inform and support the proposed project's objectives, methodologies, and expected outcomes. This analysis helps to clarify the purpose and potential of the project. Lastly, it creates a foundation for the project by demonstrating how it builds upon and contributes to the existing body of knowledge in the field.

2.1 Research Background and Related Work

2.1.1 Japanese Culture In Malaysia

According to Mamat, R., Rashid, R. A., Paee, R., & Ahmad, N. (2022) VTubers and anime culture: A case study of Japanese learners in two public universities in Malaysia, Japanese culture especially through mediums like anime, manga, music, voice actors and VTubers has become deeply ingrained in the lives of many young people in Malaysia. The results show that 74.8 % of the respondents know how to speak and write in Japanese, while only 8.7% do not know the language. Other respondents, amounting to 16.5%, know how to write and speak the language a little. This proves that the cultural fascination has sparked a growing curiosity among the youth to explore Japanese traditions, customs, and language. The popularity of these cultural elements suggests that there is a significant opportunity to engage this audience in learning more about Japan, with a particular interest in acquiring basic Japanese language skills. This trend indicates a receptive environment where educational tools, like language-learning games, can thrive by tapping into the existing enthusiasm for Japanese culture.

2.1.2 Japanese Educational Game

Various language educational apps which include Japanese have been created throughout these years mostly on mobile devices such as LingoDeer and Duolingo. On the other hand, even though there are many Japanese educational websites and games on the PC platform, players don't really prefer picking the PC as a platform for their learning as Mobile platforms are far more convenient compared to PC platforms. Moreover, the PC does not directly allow handwritten characters where Mobile allows users to learn writing Japanese characters directly. Meanwhile in the VR platform, Mondly VR and ImmerseMe VR provides a virtual conversation education for players more than focusing on the japanese characters itself. Other choices such as VR Chat allow us to choose our own conversation communities or groups such as the japanese themed word to have real life conversation between players.

2.1.3 Virtual Reality Education

Virtual reality education games have the unique ability to simulate real-life conversations, allowing users to experience the nuances of Japanese culture in a more authentic and immersive way. This immersive experience fosters a deeper understanding of the language, as users can engage in fluid conversations that closely mimic those they might encounter in real-world settings. However, while VR excels in providing these realistic conversational experiences, there remains a significant gap in the market for more foundational educational games that focus on the basics, such as recognizing Japanese characters and understanding fundamental grammar structures.

Although there is no special mention about language educational games, Oyelere, S. S., Bouali, N., Kaliisa, R., Obaido, G., Yunusa, A. A., & Jimoh, E. R. (2020). Exploring the trends of educational virtual reality games: a systematic review of empirical studies. Smart Learning Environments, 7(1) shows a list and milestones of educational VR games created by their creator and country. The potential of the VR platform in this area is vast. Unlike traditional PC-based learning, which often relies on the use of a mouse for writing, VR controllers can replicate the natural motion of handwriting, offering a more intuitive and engaging way to practise writing Japanese characters. This capability is particularly valuable for language learners, as writing by hand is a critical component of mastering the Japanese script. The VR environment, with its ability to provide a more lifelike and tactile experience, enhances the learning process by allowing users to practise writing in a manner that feels more authentic and connected to real-world use.

Moreover, VR's immersive nature makes it a compelling tool for education, especially for younger audiences who are drawn to new and innovative technologies. As VR is a relatively newer platform, it holds a certain allure for younger generations, making it an attractive medium for learning. The ability to interact with the virtual world in a more physical and engaging way can increase motivation and interest in learning Japanese, turning what might otherwise be a challenging task into an enjoyable and rewarding experience. As the VR platform continues to evolve, there is a growing opportunity to develop educational games that not only teach the basics of Japanese language and grammar but do so in a way that fully leverages the immersive and interactive potential of virtual reality.

2.1.4 Optical Character Recognition (OCR)

Optical Character Recognition (OCR) plays a crucial role in this project, serving as a core component that bridges the gap between the player's physical input and the game's ability to understand and evaluate that input. One of the most challenging aspects of developing "Hiragana Sensei" is the integration of OCR technology within the Unity game engine. This integration is not just about recognizing characters but doing so in a way that is seamless, accurate, and responsive within a real-time interactive environment.

The project's goal is to leverage OCR to accurately recognize the Hiragana characters that players draw using their VR controllers. This requires not only precise character recognition algorithms but also the ability to handle variations in handwriting styles, as each player may have a different way of forming characters. The challenge lies in ensuring that the OCR system is robust enough to correctly interpret these variations while maintaining the fluidity and engagement of the gameplay.

Integrating OCR into Unity involves several technical hurdles, including optimising the OCR algorithms for performance, ensuring compatibility with the VR interface, and creating a user experience that feels natural and intuitive. The OCR system must be able to process input in real-time, providing immediate feedback to the player, which is essential for maintaining the educational value and interactive nature of the game.

Furthermore, this project seeks to push the boundaries of what OCR can achieve within a VR context. By implementing a system that recognizes handwritten characters, "Hiragana Sensei" aims to offer a more authentic and hands-on learning experience compared to traditional typing-based methods. The successful integration of OCR will not only enhance the gameplay but also serve as a significant technological achievement within the realm of educational VR applications.

2.2 Project Background

The "Hiragana Sensei" project is designed to immerse players in a virtual environment where they can practise writing Japanese Hiragana characters using VR controllers. This innovative approach allows players to engage in a more interactive and hands-on learning experience, simulating the real-life act of handwriting in a digital space. Central to the project's functionality is the implementation of Optical Character Recognition (OCR) technology. This OCR system is specifically tailored to recognize the characters that players write using their VR controllers, ensuring that the input corresponds accurately to the Hiragana characters being requested by the game.

By integrating OCR, the project not only assesses the correctness of the player's written characters but also compares them with the intended characters in real-time. This comparison allows for immediate feedback, enabling players to correct mistakes on the spot and reinforcing their learning process. The virtual environment created by "Hiragana Sensei" thus offers a dynamic and engaging way for users to master the writing of Hiragana, providing a seamless blend of education and interactive gameplay.

2.3 Literature Review

2.3.1 Techniques

Image Preprocessing: Techniques such as noise reduction, thresholding, and edge detection are applied to clean up the input data (the drawn characters) before it is fed into the OCR algorithm for better accuracy.

Template Matching: A potential technique where the player's drawn characters are compared against a database of Hiragana templates to determine correctness.

Convolutional Neural Networks (CNNs): Often used in OCR for recognizing patterns in images, CNNs could be employed to train a model that can accurately distinguish between different Hiragana characters based on the player's input.

Supervised Learning: The model could be trained using labelled data—images of handwritten Hiragana characters paired with their corresponding labels—to improve recognition accuracy over time.

2.3.2 Algorithms

OCR Algorithms: Implemented to recognize and interpret the Hiragana characters drawn by players. This may involve machine learning models trained to identify handwritten Japanese characters, with specific focus on accuracy and real-time processing.

2.3.3 Technology

VR Controllers: Utilises VR controllers to reproduce handwriting movement in a virtual environment.

VR Development Platform: Unity3D is used as the primary game engine to build and design the virtual classroom and interactive elements. Unity's VR integration tools help manage interactions between the player and the virtual world.

2.4 Feasibility Study

2.4.1 Technical

Software and Hardware: The core technologies required for "Hiragana Sensei"—such as VR hardware, Unity3D game engine, and OCR software—are well-established and readily available. Unity3D offers support for VR development, and there are existing libraries and frameworks for integrating OCR into Unity, making the technical implementation possible.

Development Complexity: While the integration of VR and OCR presents some challenges, these can be managed with careful planning and execution. The technical skill set required includes proficiency in Unity3D, experience with VR development, and knowledge of machine learning for OCR. However, the project is within the technical capabilities of an experienced developer.

Scalability: The project can be scaled to include more complex features or additional languages as needed. The use of Unity3D allows for easy expansion and cross-platform deployment, which supports future growth and enhancements.

2.4.2 Financial

Initial Costs: The initial financial cost includes purchasing VR hardware, Unity3D licensing (if not using the free version), OCR software or libraries, and other additional tools or assets needed for development.

Ongoing Costs: Maintenance, updates, and possible future expansions will incur ongoing costs. These could include server costs if the game features online elements, or further licensing fees for advanced tools. Marketing and distribution also require budget allocation, particularly if the game is intended for a wider audience.

2.4.3 Operational

Development Timeline: Given the project's scope, it is feasible to complete development within a one to two semesters roughly around 5-6 months. This includes time for initial development, testing, and iteration based on user feedback.

Sustainability: The project is sustainable if it meets its educational goals and finds a niche within the VR educational market. Regular updates, content expansions, and responsive support will be key to maintaining user interest and ensuring the game's long-term viability.

2.5 Chapter Summary and Evaluation

In conclusion, this chapter outlines the integration of Optical Character Recognition (OCR) within the Unity framework to accurately compare Hiragana characters written by players against predefined standards. The incorporation of OCR technology into a VR environment presents several unique challenges, including ensuring that handwriting recognition is both realistic and precise, as well as seamlessly integrating this functionality within the immersive VR platform. Despite these challenges, this individual project team is committed to addressing and overcoming these obstacles to achieve a successful implementation. This chapter underscores the importance of these technical considerations and sets the stage for the continued development and refinement of the "Hiragana Sensei" project, aiming to provide an innovative and effective learning tool.

Chapter 3

Methodology and Requirements Analysis

3 Methodology and Requirements Analysis

This chapter details the methodology and steps involved in the creation of this project, focusing on the design and implementation of interactive elements and the virtual environment. It also includes an analysis of the criteria used to determine the correct writing of Japanese Hiragana characters, ensuring that the educational objectives of the game are met.

3.1 Model Methodology

This project will follow the Agile methodology due to its flexibility and adaptability. Agile is well-suited for individual development efforts like this one, where extensive testing and redesign are necessary for various interactions within the game. By using Agile, the project can effectively respond to challenges as they arise, allowing for continuous improvement and iteration throughout the development process.

Figure 3.1.1 Agile Methodology Model Diagram

3.2 Requirement Analysis

3.2.1 Development environment

The development environment for "Hiragana Sensei" will be primarily built using Unity 3D, a versatile game development engine that supports VR integration. The coding and scripting will be handled using Visual Studio, with version control managed through Git. Additionally, OCR libraries such as Tesseract OCR will be employed to recognize and interpret handwritten Hiragana characters within the game. The hardware required for development includes a high-performance PC equipped with a suitable GPU, ample RAM, and a VR headset compatible with Unity, such as the Oculus Rift. This setup will ensure that the system can handle the real-time rendering and testing required for VR development. The development environment is essential to facilitate smooth iteration and ensure that all interactive elements are properly implemented and tested.

3.2.2 Operation environment

The operational environment focuses on the end-user experience. Players will need a VR headset, such as the Oculus Rift, along with VR-compatible controllers to interact with the game. The game will be designed to run on Windows and macOS systems that support VR technology. If future updates include multiplayer or online features, a stable internet connection will be necessary. This environment will be optimised to deliver a smooth and immersive experience, with a particular emphasis on low latency and accurate OCR input recognition, ensuring that users can enjoy seamless gameplay.

3.2.3 Non-functional Requirements

Non-functional requirements focus on the quality attributes of the game. The performance of "Hiragana Sensei" is crucial, with the game needing to run smoothly at a consistent frame rate, ideally 60 FPS, to prevent motion sickness and provide a comfortable VR experience. The accuracy of the OCR system is another critical factor, ensuring that handwritten characters are recognized correctly with minimal errors. Usability is emphasised, with the interface designed to be intuitive and accessible even for players who are new to VR. The game will be reliable, with a stable system that minimises crashes and bugs, ensuring uninterrupted gameplay. Security, though less critical in an offline VR game, will be considered if any user data is involved. Finally, compatibility across various VR headsets and controllers will be ensured to maximise the game's accessibility and reach.

3.2.4 Functional requirement

In terms of functional requirements, "Hiragana Sensei" must accurately recognize Hiragana characters written by players using OCR technology. The game will allow players to interact with virtual students who ask questions in Romaji, requiring players to write the correct Hiragana character in response. A dynamic hint system will guide players, gradually reducing hints as they progress and improve. A scoring mechanism will reward correct answers and deduct points for incorrect ones, maintaining engagement. The game will also ensure a continuous learning experience with adaptive difficulty, providing an appropriate challenge level for players as they advance.

3.3 Requirement Diagram

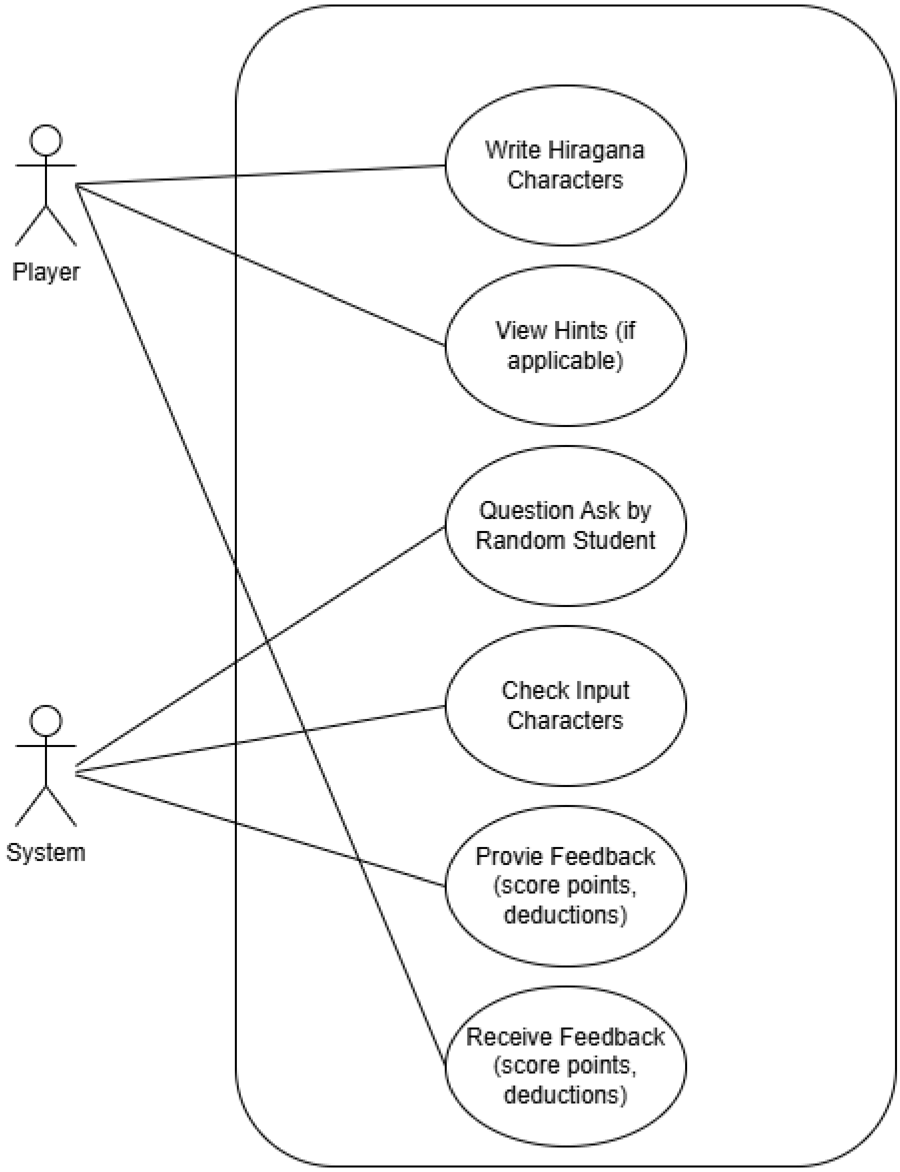


Figure 3.6.1 Use Case Diagram

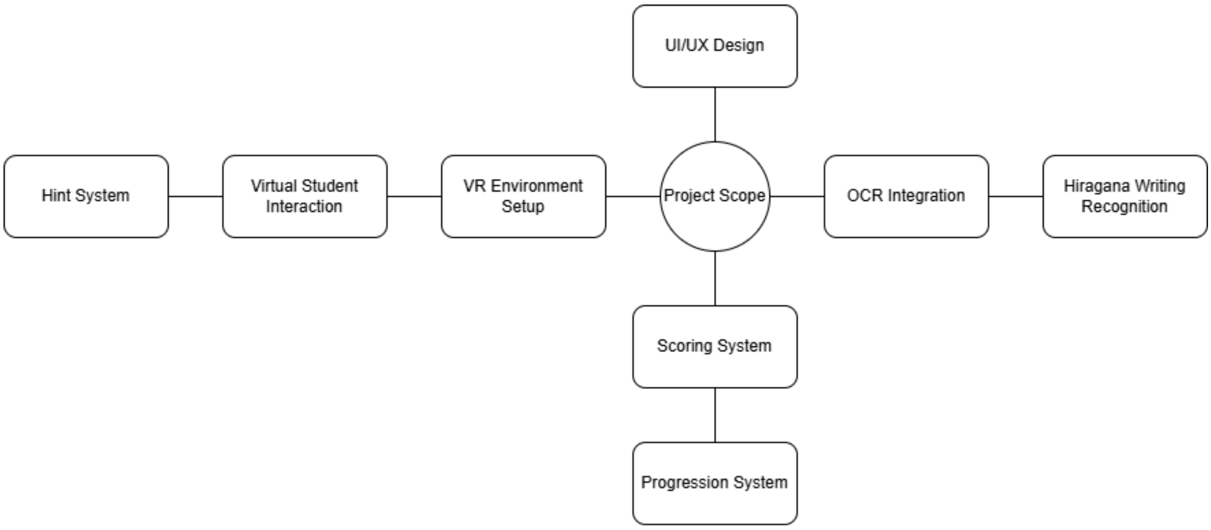


Figure 3.6.2 Project Scope Diagram

3.4 Chapter Summary and Evaluation

This chapter outlines the methodology and requirements analysis for the "Hiragana Sensei" project. The Agile model has been chosen to provide flexibility and adaptability, ensuring that the project can evolve based on feedback and testing throughout the development process. The chapter also details the development environment, which includes the tools and technologies necessary for building the game, as well as the operational environment that defines where and how the game will be used. Furthermore, the non-functional and functional requirements are clearly defined to ensure the project meets its performance goals and delivers the intended features. The use case diagram visually represents the interactions between users and the system, highlighting the key functionalities. Additionally, the project scope diagram offers an overview of the project's boundaries and components. Together, these elements provide a comprehensive understanding of the technical and operational framework guiding the project's development.

Chapter 4

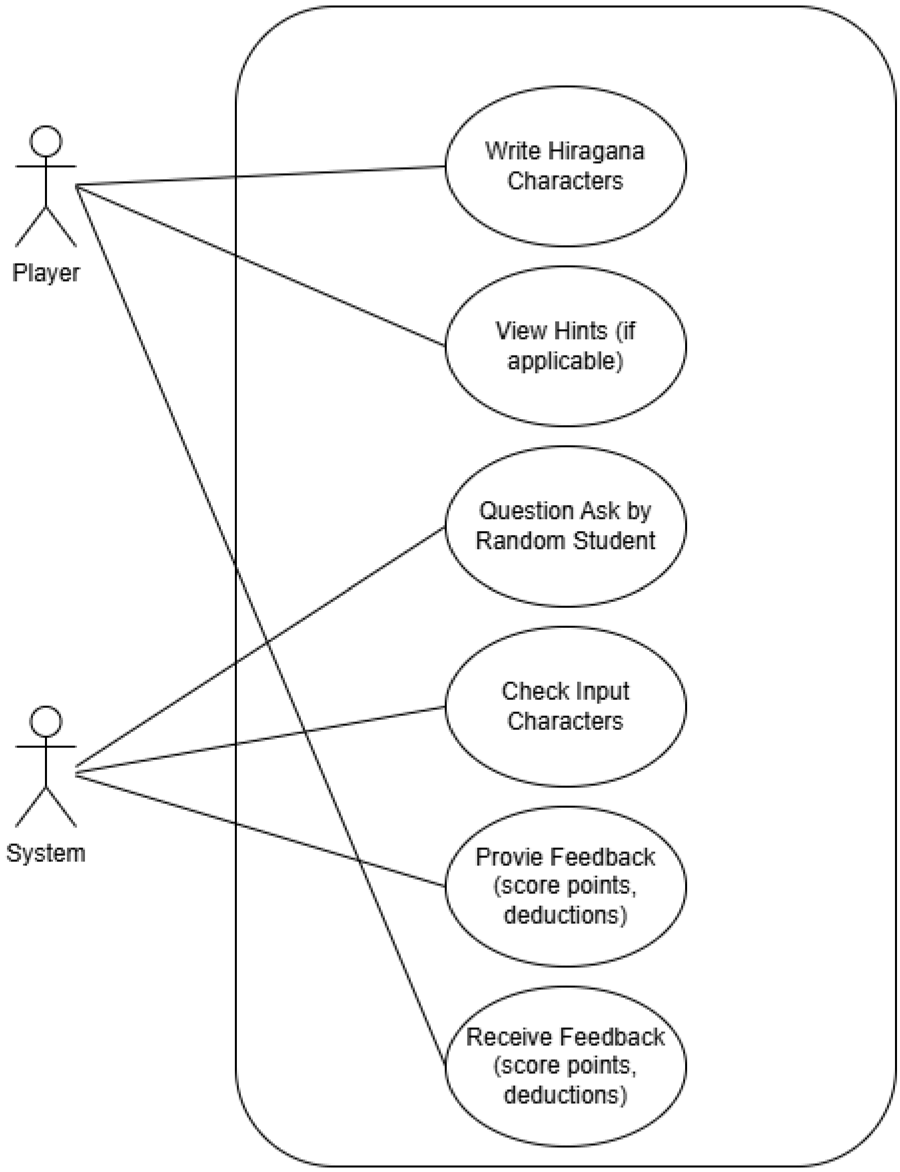
System Design

4 System Design

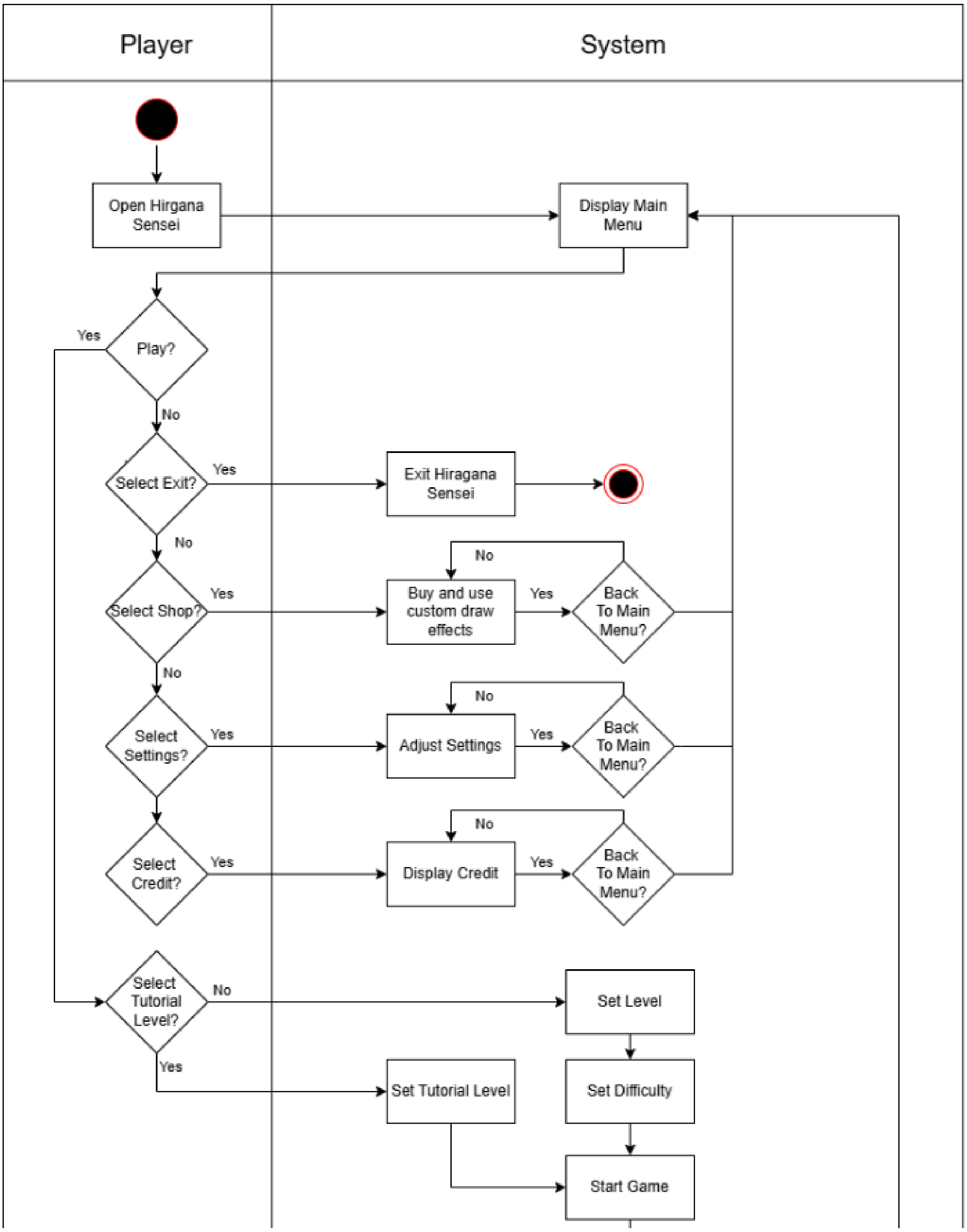
This chapter delves into the structural and functional design of the "Hiragana Sensei" game, providing a comprehensive overview of its various elements. It begins by presenting key diagrams such as use case, activity, game flow, class, sequence, and collaboration diagrams, which visually depict how the game operates and interacts with the player. These diagrams help to clarify the different actions, processes, and interactions within the game. Furthermore, the chapter covers the core functions and features of the game, highlighting how players will engage with the educational content. It also provides a detailed list of assets that will be used, including characters, environments, and objects. Finally, the chapter introduces the screen layouts, showcasing the user interface and visual design that players will interact with throughout the game.

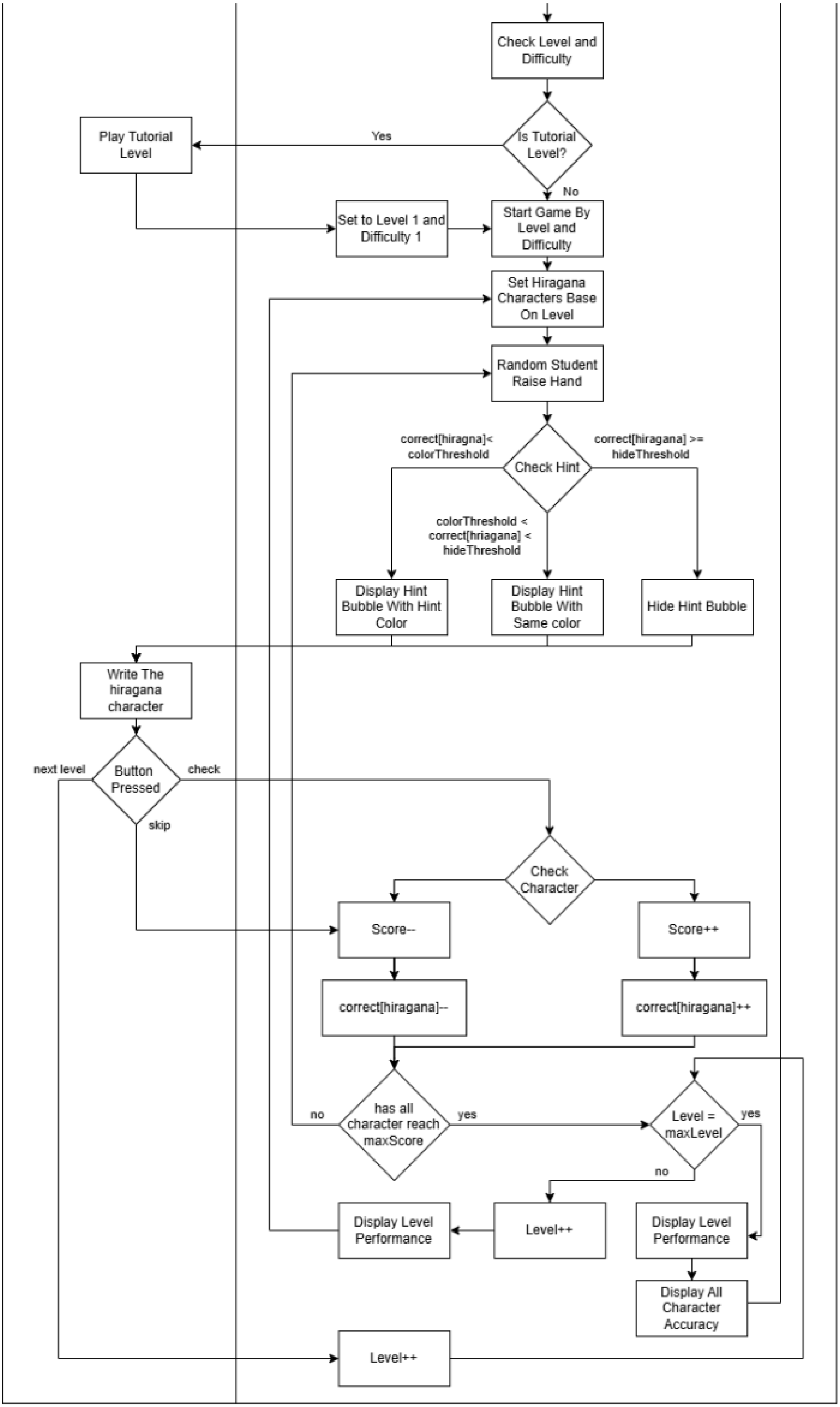
4.1 Diagrams

4.1.1 Use case Diagram

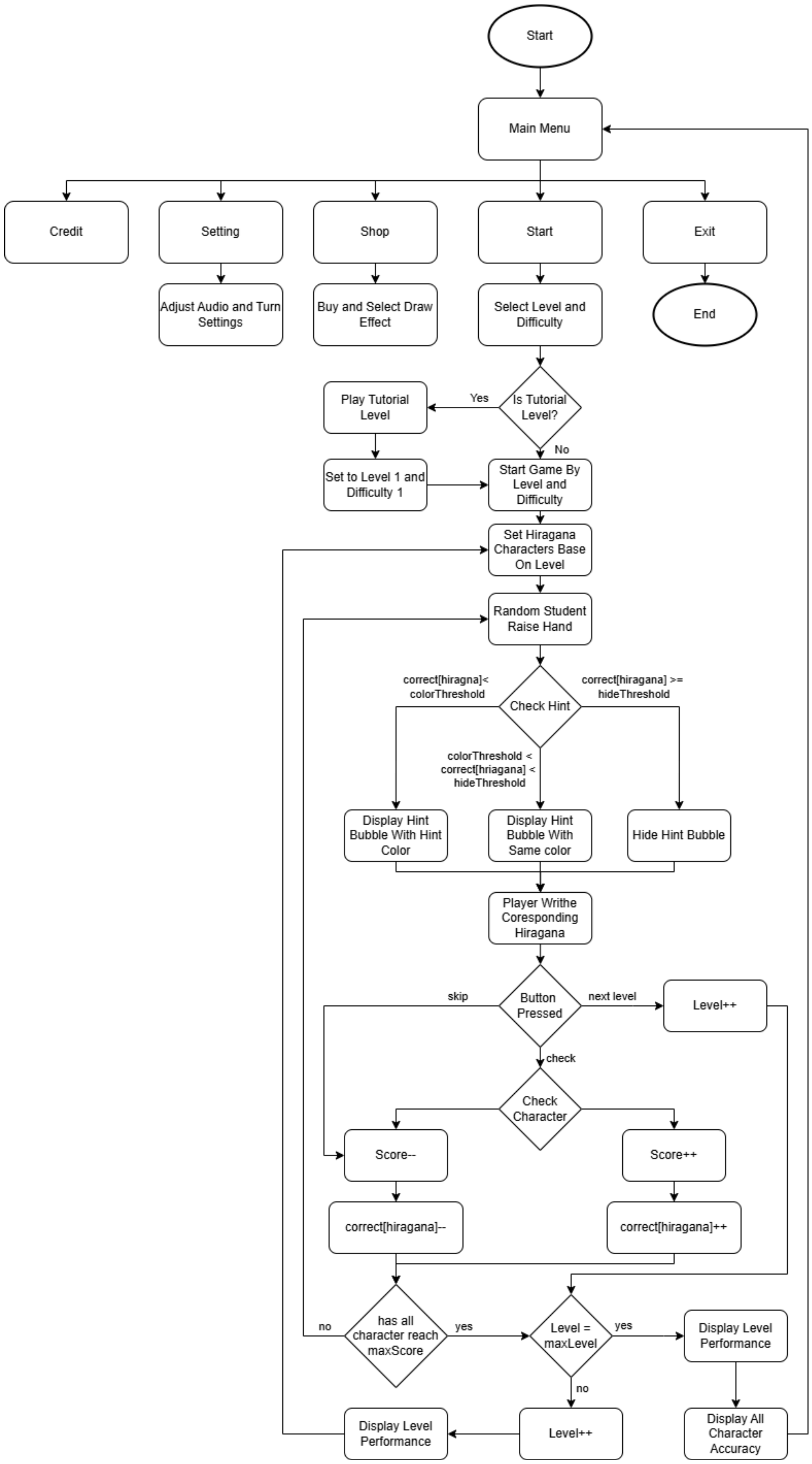


4.1.2 Activity Diagram

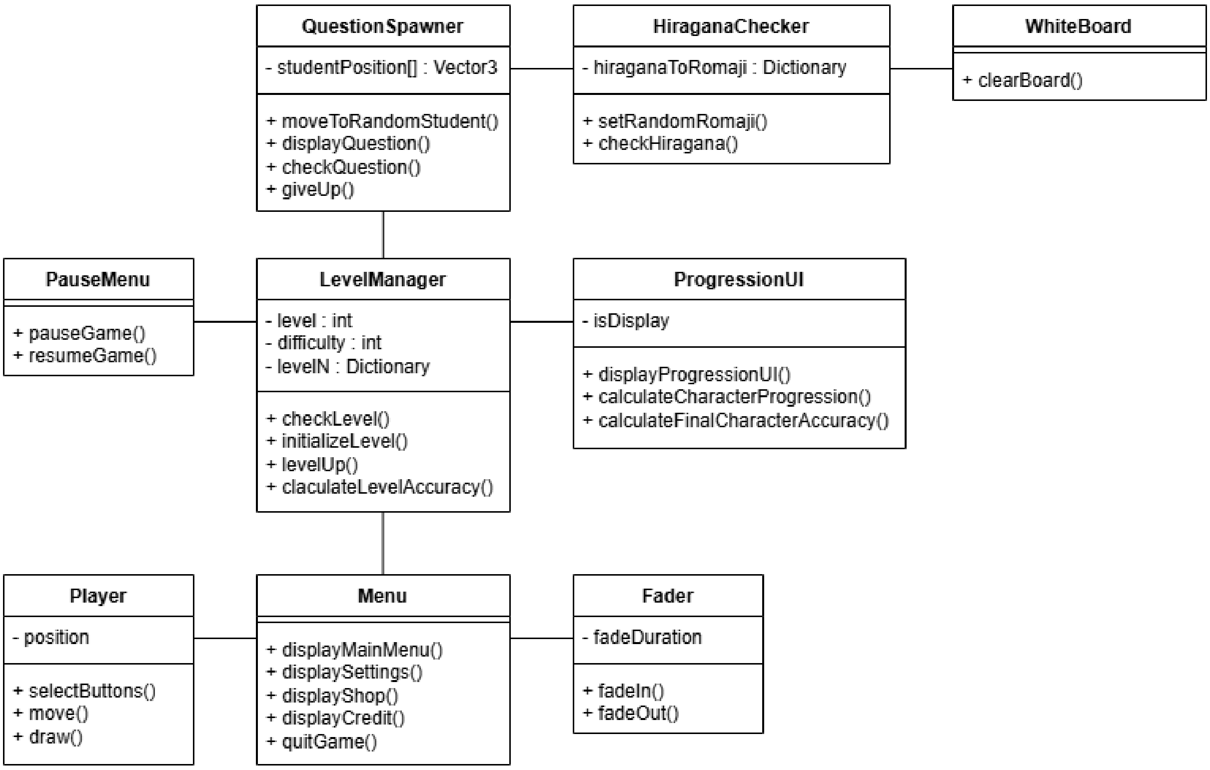



Check Level and Difficulty
Is Tutorial Level?
Yes
Play Tutorial Level
No
Set to Level 1 and Difficulty 1
Start Game By Level and Difficulty
Set Hiragana Characters Base On Level
Random Student Raise Hand
Check Hint
correct[hiragna]< colorThreshold
correct[hiragana] >= hideThreshold
colorThreshold < correct[hriagana] < hideThreshold
Display Hint Bubble With Hint Color
Display Hint Bubble With Same color
Hide Hint Bubble
Write The hiragana character
Button Pressed
next level
check
skip
Check Character
Score--
Score++
correct[hiragana]--
correct[hiragana]++
has all character reach maxScore
no
yes
Level = maxLevel
yes
no
Display Level Performance
Level++
Display Level Performance
Display All Character Accuracy
Level++

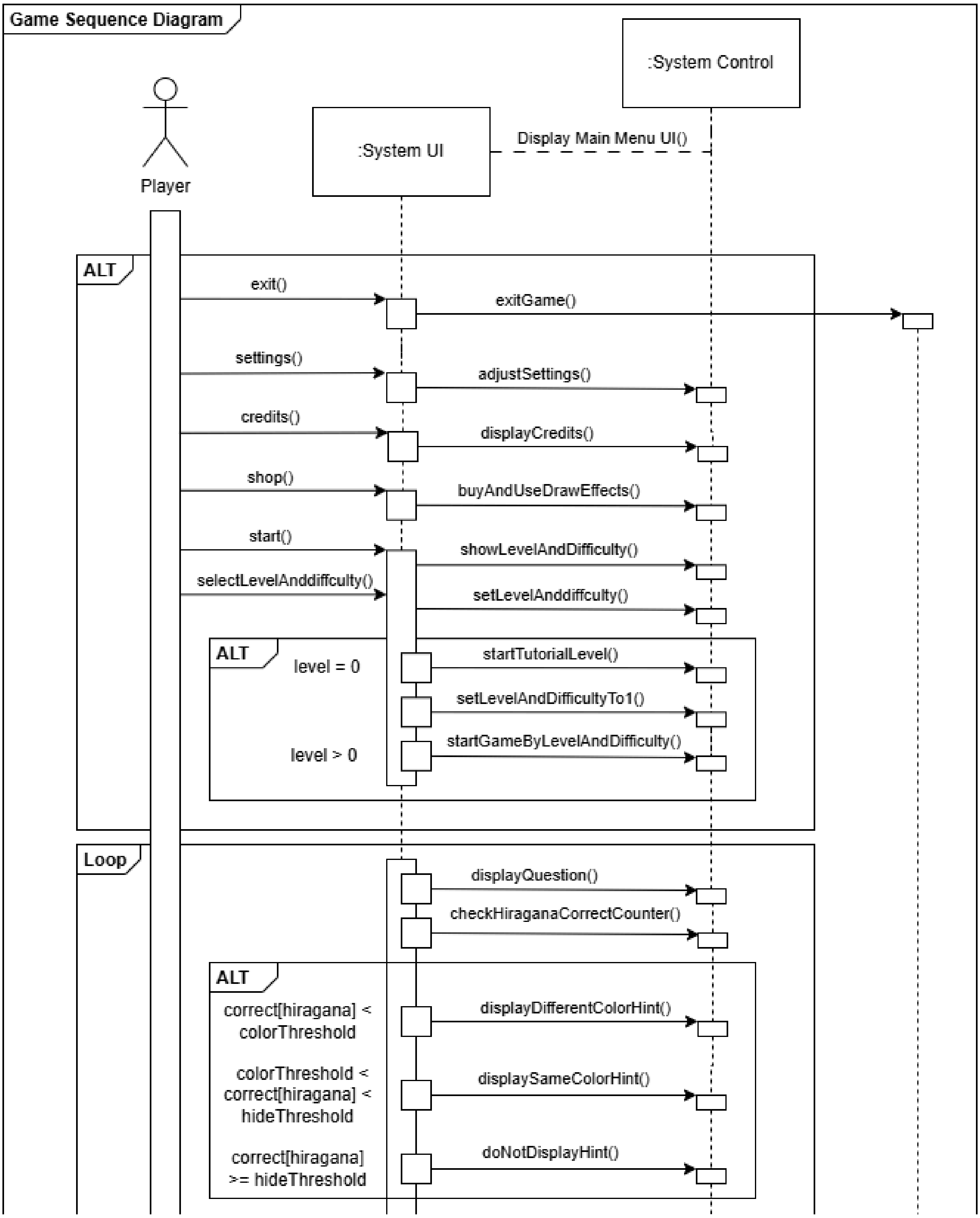
4.1.3 Game Flow Diagram

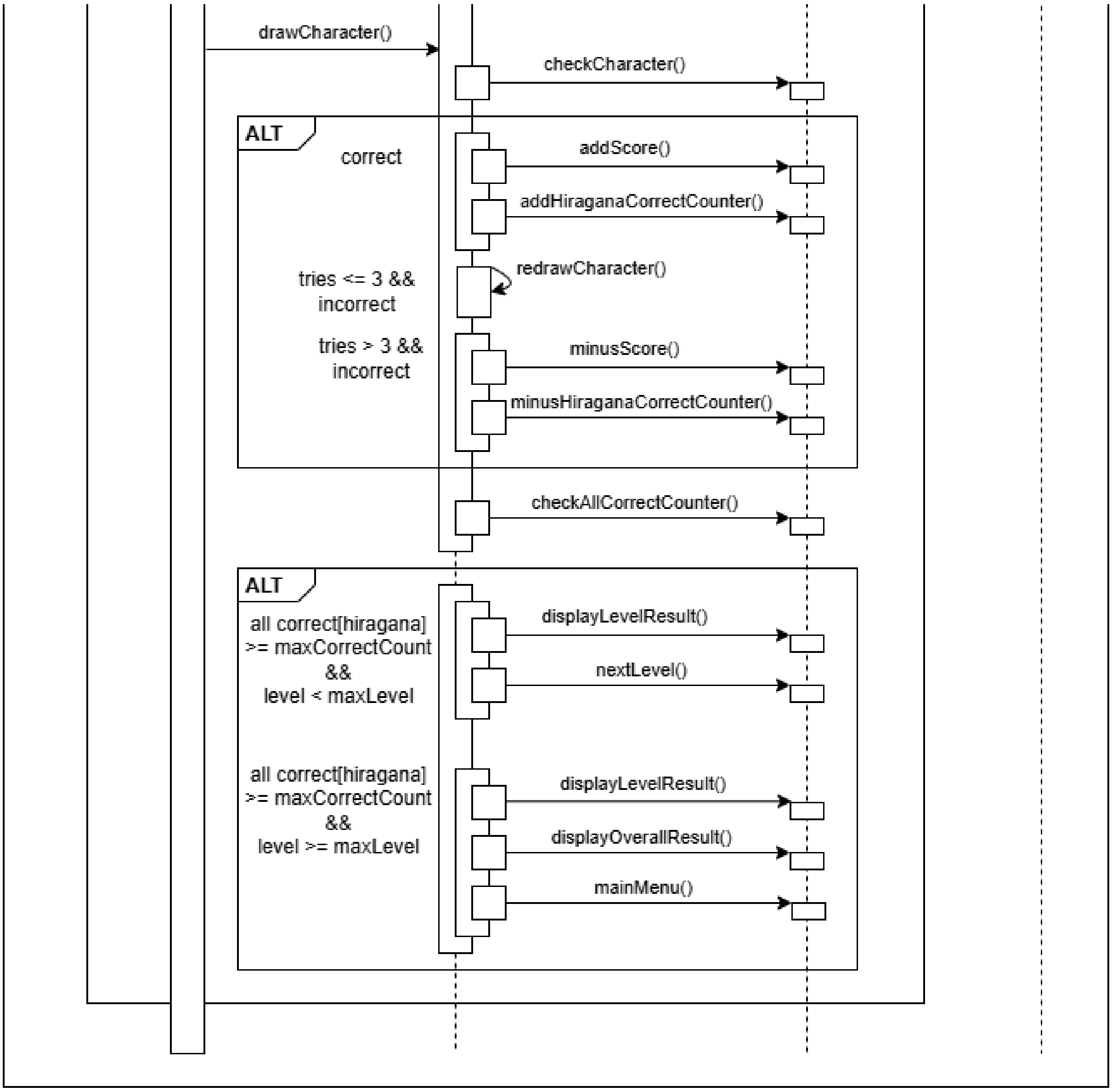


4.1.4 Class Diagram

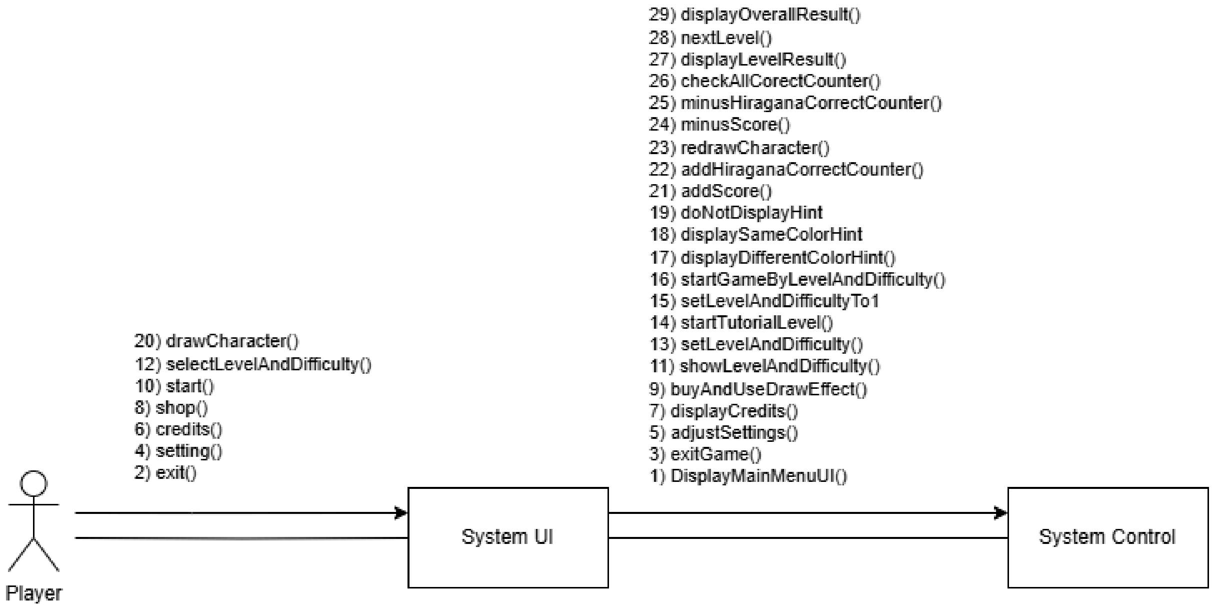


4.1.5 Sequence Diagram




drawCharacter()
checkCharacter()
ALT
correct
addScore()
addHiraganaCorrectCounter()
redrawCharacter()
tries <= 3 &&
incorrect
tries > 3 &&
incorrect
minusScore()
minusHiraganaCorrectCounter()
checkAllCorrectCounter()
ALT
all correct[hiragana]
>= maxCorrectCount
&&
level < maxLevel
displayLevelResult()
nextLevel()
all correct[hiragana]
>= maxCorrectCount
&&
level >= maxLevel
displayLevelResult()
displayOverallResult()
mainMenu()

4.1.6 Collaboration Diagram



4.2 Functions & Features

4.2.1 Immersive VR Classroom

The game offers an engaging virtual environment designed to resemble a lively Japanese classroom. Players will find themselves surrounded by animated student avatars, each with distinct expressions and movements, simulating the atmosphere of a real classroom. This immersive setting not only enhances the learning experience by making it more engaging but also helps players feel like they are truly part of a teaching environment, bridging the gap between virtual and real-life interaction.

4.2.2 Interactive Teaching

In "Hiragana Sensei," players take on the role of a teacher, interacting directly with student avatars. As the game progresses, students will raise their hands and ask for the Hiragana equivalent of a given Romaji (Japanese written in English letters). The player selects a student by pointing with the VR controller, creating an interactive experience that mirrors real-life classroom engagement. This interaction reinforces the player's role as a teacher and deepens their involvement in the learning process.

4.2.3 Hint System

To support the learning curve, a hint system is integrated into the game. Upon selecting a student, five hint bubbles will appear around the student, offering visual clues to help the player identify the correct Hiragana character. These hints are especially useful for new players still familiarizing themselves with the characters, and as players improve, the hints are gradually reduced to challenge them further. This feature not only aids in initial learning but also gradually encourages independent recall.

4.2.4 Writing and Drawing

A key feature of "Hiragana Sensei" is the ability to write and draw Hiragana characters using the VR controllers. Players are required to physically write the correct Hiragana character in response to the student’s query, promoting active learning. This hands-on approach reinforces memory retention and helps players better connect the written characters with their Romaji counterparts. The act of drawing characters adds a kinesthetic element to the learning process, making it more interactive and effective.

4.2.5 Input Recognition

Optical Character Recognition (OCR) plays a crucial role in the game's functionality, as it allows the game to recognize the player's written input. When the player writes or draws the Hiragana character using the VR controllers, the OCR system compares the input with the expected character. This real-time recognition ensures that the player's writing is evaluated accurately, providing immediate feedback. The inclusion of OCR technology enhances the game’s educational value by allowing players to practice writing with precision and receive instant assessment of their progress.

4.3 Asset List

4.3.1 VR hands

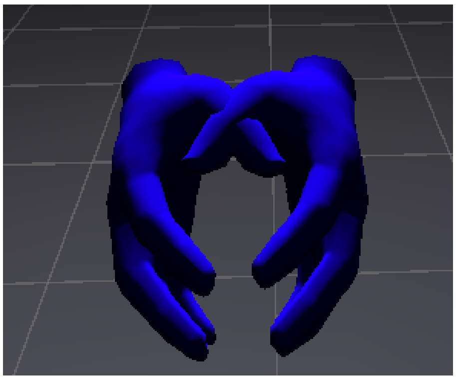


Figure 4.3.1.1

In this project, I will implement a basic VR hand input system shown in figure 4.3.1.1, utilising the default VR hands asset provided by Unity. This method allows for efficient and straightforward hand-tracking functionality within the game, enabling players to interact with the virtual environment by selecting students and writing Hiragana characters using VR controllers. By following this accessible approach, I ensure that the input system remains functional and user-friendly while maintaining focus on the educational objectives of the project.

4.3.2 Main Menu

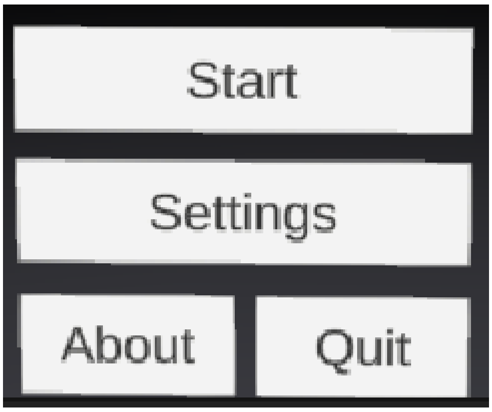


Figure 4.3.2.1

In this project, I will integrate a simple VR menu interface shown in figure 4.3.2.1, developed following a tutorial from YouTube with some custom modifications with some extra functions and better looking. The interface will include essential options such as start, settings, shop, about, and exit, providing users with a straightforward and intuitive navigation system. This approach allows for ease of use and helps maintain focus on the educational gameplay. You can access the source of this menu design here.

4.3.3 Male Students

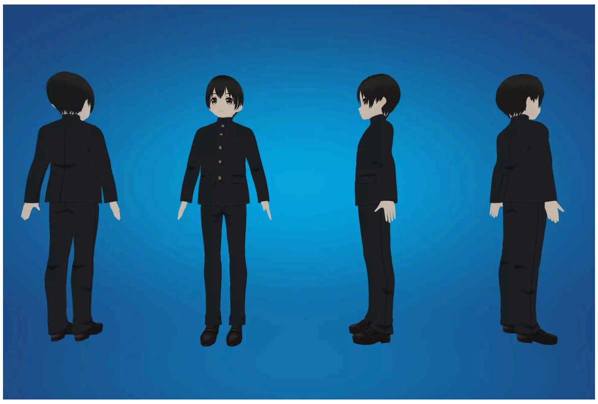


Figure 4.3.3.1

This project will use highschool male character assets, as depicted in Figure 4.3.3.1. This character is styled in anime-like designs by wearing a Japanese high school uniform outfit. Some custom animation such as sitting and asking questions will be added into the character.

4.3.4 Female Students

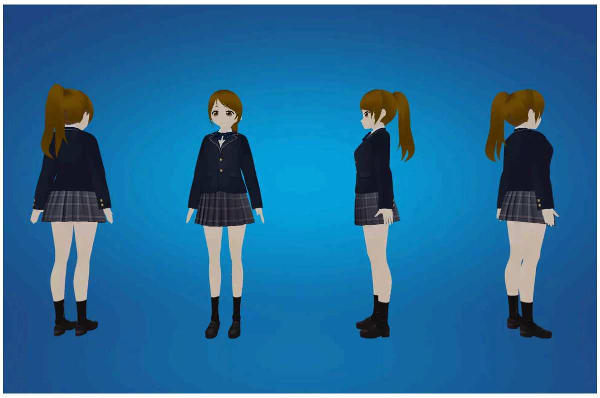


Figure 4.3.4.1

This project will use highschool female character assets, as depicted in Figure 4.3.3.1. This character is styled in anime-like designs by wearing a Japanese high school uniform outfit. Some custom animation such as sitting and asking questions will be added into the character.

4.3.5 Classroom

Figure 4.3.5.1

Figure 4.3.5.1 represents the main environment for this game. It showcases a virtual classroom designed to accommodate 25 students, arranged in a 5x5 layout. This classroom setting will serve as the primary platform where the player interacts with students and engages in the learning experience, making it central to the gameplay.

4.4 Concept Art

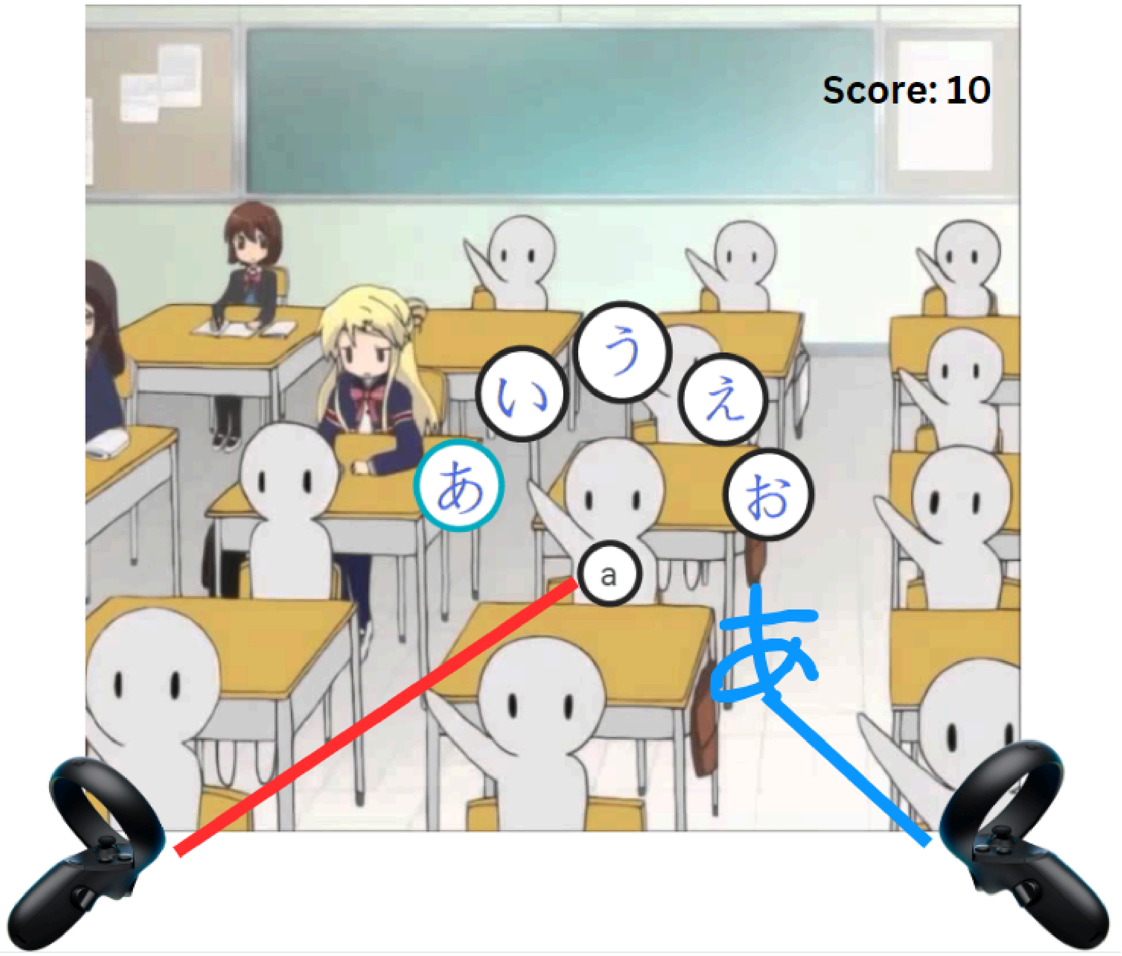


Figure 4.4.1 Concept Art for First Person View of Gameplay

Figure 4.4.1 provides a detailed visual representation of the gameplay concept for "Hiragana Sensei." This concept art illustrates the key elements of the virtual classroom environment, where players interact with virtual students while using VR controllers to write Japanese Hiragana characters. The image highlights the immersive experience the game offers, showcasing how students raise their hands to ask for the corresponding Hiragana of a given Romaji, and how the player selects the student and receives visual hints. The concept art also emphasizes the user interface, hint system, and how the player's actions are integrated into the educational flow of the game. This visual serves as a foundation for understanding the design and interactive elements of the gameplay.

4.5 Level Design & In Game Design

The game will function as depicted in Figure 4.4.1, the concept art, showcasing the interactive gameplay of "Hiragana Sensei." In the game, the player assumes the role of a teacher in a virtual classroom setting. The students, represented by virtual characters, will raise their hands to ask questions in Romaji (English characters), and it is the player’s responsibility to answer the student who posed the question. Players must use the VR controllers to write the corresponding Japanese Hiragana characters on the whiteboard to answer the question. The process of selecting students and writing characters is designed to simulate an authentic classroom experience, allowing the player to engage in active teaching and learning. As the first few times the character is asked by the student, hints will be provided in the form of visual bubbles, gradually reducing as the player answers the character correctly. This gameplay aims to ensure a smooth and enjoyable learning curve while fostering the player's ability to both recognize and write Hiragana characters effectively.

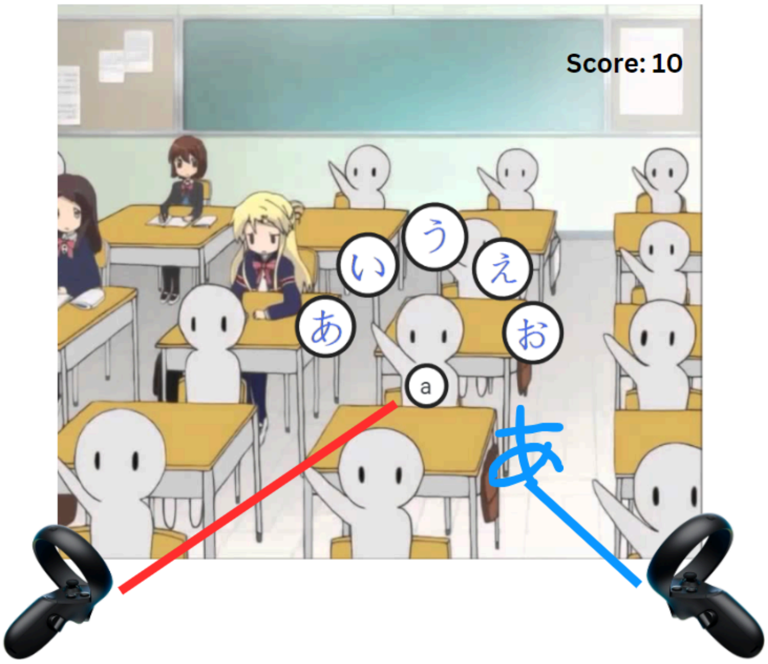


Figure 4.5.1 2nd Difficulty

Figure 4.5.1 shows the increase of difficulty level. As the player continues to answer correctly, the game's difficulty increases by reducing the assistance provided. Initially, colorful circular hints appear around the students to guide the player in identifying the correct Hiragana characters. However, as the player becomes more proficient, these hints will be gradually removed, forcing the player to rely on their memory and knowledge of the characters. This system not only enhances the challenge but also encourages the player to master the recognition and writing of Hiragana without external aids, further improving their learning experience.

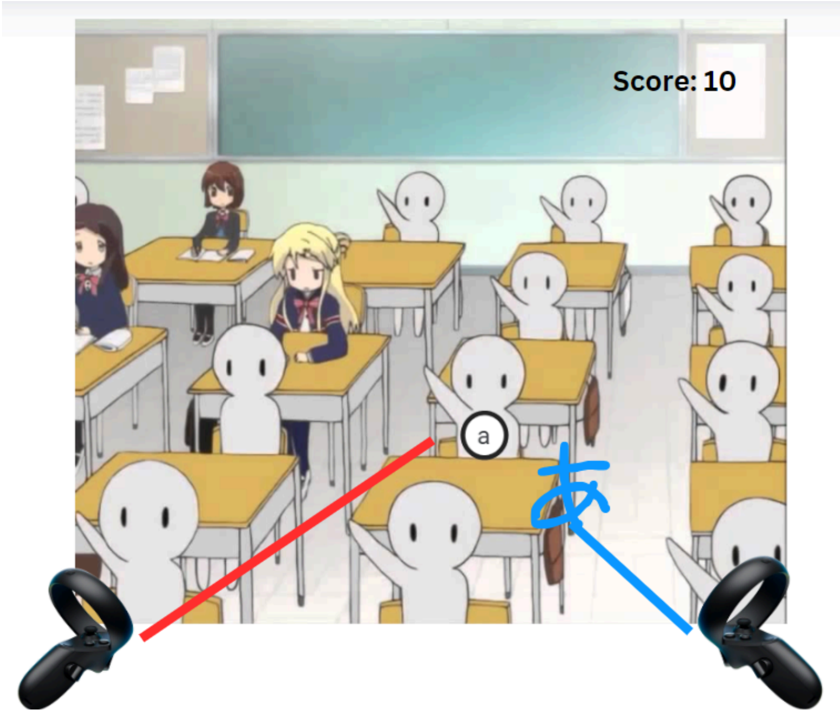


Figure 4.5.2 3rd Difficulty

Figure 4.5.2 shows the final difficulty. As the player continues to answer correctly, the difficulty increases even further. Eventually, the hint circles will disappear, requiring the player to write the correct Hiragana character without any visual references. This progression challenges the player's mastery of Hiragana, pushing them to rely entirely on their memory and understanding. The absence of hints adds an extra layer of difficulty, ensuring that the player can accurately recall and write the characters, reinforcing their learning and retention in a more advanced, independent manner.

Figure 4.4.3 Hiragana Character List

This project aims to further increase the difficulty by introducing new rows of Hiragana characters, as shown in Figure 4.4.3. After the player achieves a certain number of correct answers, additional characters will be progressively introduced, enhancing the challenge and expanding the player's learning scope. If feasible within the project’s timeline, this feature will ensure that players are continually exposed to new characters, promoting a deeper understanding of the full range of Hiragana. This gradual introduction of new characters adds complexity, making the learning process more dynamic and comprehensive.

4.6 Screen Layout

4.6.1 Home Screen

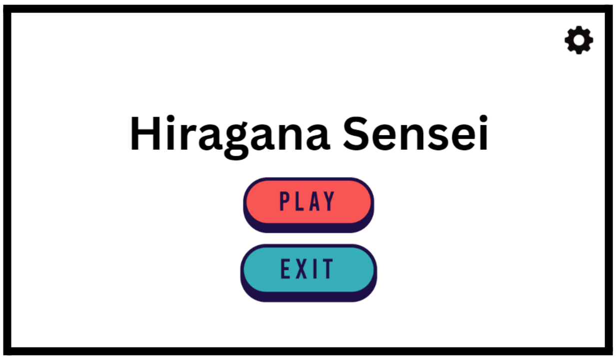


Figure 4.6.1.1 Home Screen

Figure 4.5.1.1 illustrates the home screen layout for "Hiragana Sensei." Since the game focuses on a streamlined and simple experience, the home screen is designed with minimal elements. It features three primary buttons: Start, Exit, and Settings, along with the game title prominently displayed. The simplicity ensures an easy-to-navigate interface, prioritizing functionality over unnecessary complexity. Although a background will eventually be applied to enhance the visual appeal, it is not included in the current screen layout preview shown in this figure. This straightforward design keeps the focus on the core gameplay experience.

4.6.2 Setting Screen

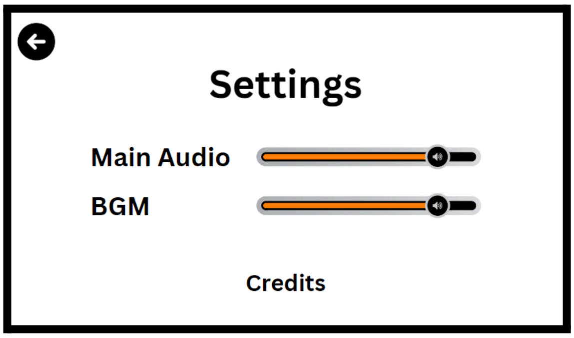


Figure 4.6.2.1 Setting Screen

In the settings menu, players will have access to audio settings, allowing them to adjust the game's volume levels. Additionally, a credits section will be included to acknowledge the individuals involved in the development of "Hiragana Sensei." The user interface will be kept simple and clean, ensuring that players can easily navigate and make adjustments without unnecessary distractions. This minimalist design approach aligns with the overall streamlined nature of the game, maintaining focus on the educational and immersive aspects of the gameplay.

4.6.3 Gameplay Screen

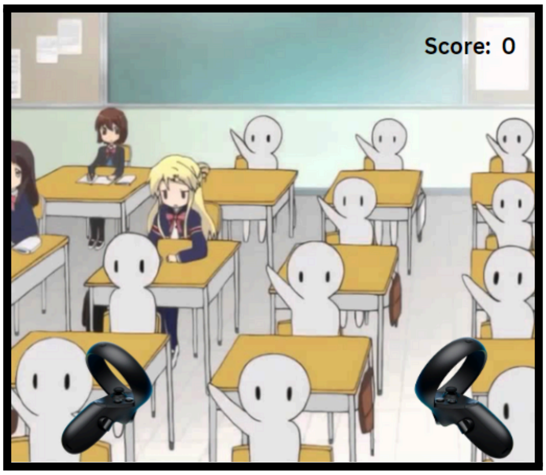


Figure 4.6.3.1 Game Play Screen

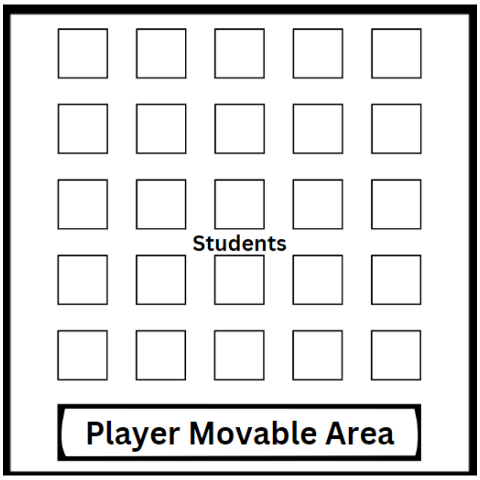


Figure 4.6.3.2 Game Play Layout

In the gameplay scene, the player assumes the role of a teacher, positioned at the front of a virtual classroom filled with 25 students arranged in a 5x5 seating format. The player's score is prominently displayed in the top right corner of the screen, providing real-time feedback on their performance. Utilizing VR controllers, players can easily select students who raise their hands to pose questions, creating an interactive and engaging teaching environment. Once a student is chosen, the player writes the appropriate Hiragana answer using the controller, as depicted in Figure 4.5 under Level Design & In-Game Design. To facilitate smooth gameplay, the player is provided with a limited movement area, allowing them to adjust their position without obstructing the view or access to the students, thereby enhancing the overall immersive experience.

Chapter 5

Implementation and Testing

5 Implementation and Testing

This chapter details the implementation and testing of the core mechanics of Hiragana Sensei. The implementation section outlines the technical components and functionality that drive the game, including writing mechanics, the Tesseract OCR wrapper and driver, Hiragana setup, question spawning, character checking, progression tracking, and performance display. Each feature is supported by detailed code snippets and explanations to provide insight into the development process. The chapter also focuses on the testing strategy, highlighting methods such as Unity-based testing, beta testing, and both black-box and white-box testing. The testing ensures the game's mechanics, performance, and user experience are thoroughly evaluated. Finally, the chapter includes a comprehensive test plan and test cases that cover key aspects of the game. These tests focus on areas such as VR interactions, UI functionality, drawing mechanics, and gameplay progression through various levels, including Level 1, Level 2, Level 3, and an extra level without hints. Additionally, the testing evaluates visual effects like particle effects to enhance the drawing experience. This structured approach ensures Hiragana Sensei meets its educational and gameplay objectives effectively.

5.1 Implementation & Coding

This section describes the development and integration of the core components of "Hiragana Sensei." The implementation involves creating an interactive writing mechanic to capture and evaluate player-drawn hiragana, utilizing Tesseract OCR for character recognition. A Hiragana checker validates player responses for accuracy, while the question spawner dynamically generates and displays questions to maintain engagement. Additionally, a performance display system provides real-time feedback, showcasing player progress and encouraging continuous improvement. These features collectively enhance the educational and interactive aspects of the game.

5.1.1 Writing / Drawing

```
void Update()
{
    isDrawing = grab.action.IsPressed();
    if (isDrawing)
    {
        Draw();
        checkDrawing = true;
    }
```

Figure 5.1.1.1 Code 1.1

```
if (Physics.Raycast(raycastOrigin, transform.forward, out _touch, _rayLength))
{
    if (_touch.transform.CompareTag("Whiteboard"))
    {
        if (_whiteboard == null)
        {
            _whiteboard = _touch.transform.GetComponent<Whiteboard>();
        }
```

Figure 451.1.2 Code 1.2

The HandDraw script allows player to perform when the player presses the grip button while the raycast on the hand points towards the “Whiteboard” tag.

```
_touchPos = new Vector2(_touch.textureCoord.x, _touch.textureCoord.y);

// Calculate pixel position
var x = (int)(_touchPos.x * _whiteboard.textureSize.x - (_drawSizeX / 2));
var y = (int)(_touchPos.y * _whiteboard.textureSize.y - (_drawSizeZ / 2));

// Prevent writing outside bounds
if (y < 0 || y > _whiteboard.textureSize.y || x < 0 || x > _whiteboard.textureSize.x) return;

if (_isFirstTouch)
{
    _lastTouchPos = new Vector2(x, y);
    _isFirstTouch = false;
}

if (_touchedLastFrame)
{
    DrawLineBresenham((int)_lastTouchPos.x, (int)_lastTouchPos.y, x, y);
    _whiteboard.texture.Apply();
}

_lastTouchPos = new Vector2(x, y);
_touchedLastFrame = true;
```

Figure 5.1.1.3 Code 1.3

This code handles drawing on a whiteboard based on the position of the raycast that is touching the whiteboard texture. It calculates the touch position on the whiteboard texture, ensuring the drawing stays within bounds. If it's the first touch, it initializes the starting point and on subsequent touches, it uses Bresenham's algorithm to draw a line between the previous and current touch positions to create lines and update the whiteboard texture after each frame.

```
private void DrawLineBresenham(int x0, int y0, int x1, int y1)
{
    int dx = Mathf.Abs(x1 - x0);
    int dy = Mathf.Abs(y1 - y0);
    int sx = x0 < x1 ? 1 : -1;
    int sy = y0 < y1 ? 1 : -1;
    int err = dx - dy;

    Color[] colors = Enumerable.Repeat(markerColor, _drawSizeX * _drawSizeZ).ToArray();

    while (true)
    {
        // Draw at the calculated position
        _whiteboard.texture.SetPixels(x0, y0, _drawSizeX, _drawSizeZ, colors);

        // Stop if the end point is reached
        if (x0 == x1 && y0 == y1) break;

        int e2 = 2 * err;
        if (e2 > -dy)
        {
            err -= dy;
            x0 += sx;
        }
        if (e2 < dx)
        {
            err += dx;
            y0 += sy;
        }
    }
}
```

Figure 5.1.1.4 Code 1.4

This method draws a line between two points on a texture using Bresenham's line algorithm. It calculates the direction and error terms to determine the next pixel to color, iterating until the end point is reached. At each step, it sets a block of pixels (based on _drawSizeX and _drawSizeZ) to the specified markerColor, creating a continuous line on the whiteboard texture. This algorithm proves a smoother line compared to normal initialization of lines.

```
if (!drawingParticles[particleIdx].isPlaying && particleIdx > 0)
{
    drawingParticles[particleIdx].Play();
}
```

Figure 5.1.1.5 Code 1.5

Additionally, some draw effects are added when the player is drawing on the whiteboard. A particle system of the selected effect will play when the player is drawing on the whiteboard.

```
if (_isTouching)
{
    if (isDrawing)
    {
        HandVibration();
    }
}
```

Figure 5.1.1.6 Code 1.6

```
private void HandVibration()
{
    // Only vibrate at intervals
    if (_vibrationTimer <= 0 && _isTouching)
    {
        controller?.SendHapticImpulse(0.5f, vibrationInterval);
        _vibrationTimer = vibrationInterval; // Reset timer
    }

    // Decrease timer
    _vibrationTimer -= Time.deltaTime;
}
```

Figure 5.1.1.7 Code 1.7

Lastly, Haptic feedback is incorporated through controller vibrations while the player draws on the whiteboard, enhancing the drawing experience and providing better tactile response.

```
// Check if drawing just ended this frame
if (checkDrawing && !isDrawing)
{
    hiraganaChecker?.CheckDrawing();
    checkDrawing = false;
    drawingParticles[particleIdx].Stop();
}
```

Figure 5.1.1.8 Code 1.8

The drawing will only be checked once the player stops drawing. This implementation ensures a better performance as direct manipulation on texture with constant calculation and color changing will cause lag if it is choking on every frame.

5.1.2 Tesseract Optical Character Recognition (OCR)

```
public string Recognize(Texture2D texture)
{
    if (_tessHandle.Equals(IntPtr.Zero))
        return null;

    _highlightedTexture = texture;

    int width = _highlightedTexture.width;
    int height = _highlightedTexture.height;
    Color32[] colors = _highlightedTexture.GetPixels32();
    int count = width * height;
    int bytesPerPixel = 4;
    byte[] dataBytes = new byte[count * bytesPerPixel];
    int bytePtr = 0;

    for (int y = height - 1; y >= 0; y--)
    {
        for (int x = 0; x < width; x++)
        {
            int colorIdx = y * width + x;
            dataBytes[bytePtr++] = colors[colorIdx].r;
            dataBytes[bytePtr++] = colors[colorIdx].g;
            dataBytes[bytePtr++] = colors[colorIdx].b;
            dataBytes[bytePtr++] = colors[colorIdx].a;
        }
    }
```

Figure 5.1.2.1 Code 2.1

The most important method in the TesseractWrapper script is the Recognize method, this method first validates that the Tesseract OCR engine is initialized using _tessHandle. Then, it prepares the image data for Tesseract by converting the pixel colors from the texture into a byte array. This byte array represents the RGBA values of each pixel, arranged bottom-to-top to align with Tesseract's input requirements.

```
IntPtr imagePtr = Marshal.AllocHGlobal(count * bytesPerPixel);
Marshal.Copy(dataBytes, 0, imagePtr, count * bytesPerPixel);

TessBaseAPISetImage(_tessHandle, imagePtr, width, height, bytesPerPixel, width * bytesPerPixel);

if (TessBaseAPIRecognize(_tessHandle, IntPtr.Zero) != 0)
{
    Marshal.FreeHGlobal(imagePtr);
    return null;
}

IntPtr confidencesPointer = TessBaseAPIAllWordConfidences(_tessHandle);
int i = 0;
List<int> confidence = new List<int>();

while (true)
{
    int tempConfidence = Marshal.ReadInt32(confidencesPointer, i * 4);

    if (tempConfidence == -1) break;

    i++;
    confidence.Add(tempConfidence);
}

int pointerSize = Marshal.SizeOf(typeof(IntPtr));
IntPtr intPtr = TessBaseAPIGetWords(_tessHandle, IntPtr.Zero);
Boxa boxa = Marshal.PtrToStructure<Boxa>(intPtr);
Box[] boxes = new Box[boxa.n];

for (int index = 0; index < boxes.Length; index++)
{
    if (confidence[index] >= MinimumConfidence)
    {
        IntPtr boxPtr = Marshal.ReadIntPtr(boxa.box, index * pointerSize);
        boxes[index] = Marshal.PtrToStructure<Box>(boxPtr);
        Box box = boxes[index];
        DrawLines(_highlightedTexture,
            new Rect(box.x, _highlightedTexture.height - box.y - box.h, box.w, box.h),
            Color.green);
    }
}
```

Figure 5.1.2.2 Code 2.2

The method then uses Marshal.AllocHGlobal to allocate memory for the image data and assigns it to Tesseract using TessBaseAPISetImage. Next, the OCR process is executed with TessBaseAPIRecognize. If recognition is successful, confidence values for the recognized words are retrieved using TessBaseAPIAllWordConfidences, while bounding box data for each word is obtained via TessBaseAPIGetWords. The confidence values are compared against a predefined MinimumConfidence threshold. For words meeting this threshold, the bounding boxes are extracted and drawn on the texture using DrawLines, highlighting recognized text with green rectangles for visual feedback.

```
        IntPtr stringPtr = TessBaseAPIGetUTF8Text(_tessHandle);
        Marshal.FreeHGlobal(imagePtr);
        if (stringPtr.Equals(IntPtr.Zero))
            return null;

#if UNITY_EDITOR_WIN || UNITY_STANDALONE_WIN
        string recognizedText = Marshal.PtrToStringAnsi (stringPtr);
#else
        string recognizedText = Marshal.PtrToStringAuto(stringPtr);
#endif

        // Filter out only Hiragana characters from the recognized text
        string hiraganaText = FilterHiragana(recognizedText);

        TessBaseAPIClear(_tessHandle);
        TessDeleteText(stringPtr);

        return hiraganaText;
```

Figure 5.1.2.3 Code 2.3

Finally, the recognized text is retrieved as a UTF-8 string with TessBaseAPIGetUTF8Text. This text is filtered to isolate Hiragana characters using the FilterHiragana function, ensuring that only the relevant script is returned. After processing, resources such as allocated memory and Tesseract's internal data are freed to prevent memory leaks. The method returns the filtered Hiragana text, making it suitable for applications like handwriting recognition or language learning tools.

5.1.3 Hiragana Setup

```
if (_tesseract.Init("jpn", datapath))
{
    Debug.Log("Tesseract Tessdata Init Successful");
    _tesseract.SetWhitelist(whitelist);
    onSetupComplete?.Invoke();
}
```

Figure 5.1.3.1 Code 3.1

The TesseractDriver script acts as a controller for the TesseractWrapper script. This code initializes the Tesseract OCR engine with the Japanese language data ("jpn") located in the specified datapath. If the initialization is successful, it configures the OCR engine to recognize only characters included in a specified whitelist, ensuring the recognition process is tailored to the desired character set. It is used to prioritize characters to be recognized on each level as well as preventing other unrelated characters from being recognized first. This segment of code determines the language to be used in the game therefore modifying the “jpn” to other languages tessdata file directly changes the checking of the whole game. This implementation makes the structure of the game very customizable and easy for modification.

```
public Dictionary<string, int> correctAttempts = new Dictionary<string, int>(); // Correct attempts
public Dictionary<string, int> incorrectAttempts = new Dictionary<string, int>(); // Incorrect attempts
public Dictionary<string, int> giveUpAttempts = new Dictionary<string, int>(); // Incorrect attempts

public readonly Dictionary<string, string> romajiToHiragana = new Dictionary<string, string>
{
    { "a", "あ" }, { "i", "い" }, { "u", "う" }, { "e", "え" }, { "o", "お" },
    { "ka", "か" }, { "ki", "き" }, { "ku", "く" }, { "ke", "け" }, { "ko", "こ" },
    { "sa", "さ" }, { "shi", "し" }, { "su", "す" }, { "se", "せ" }, { "so", "そ" },
    { "ta", "た" }, { "chi", "ち" }, { "tsu", "つ" }, { "te", "て" }, { "to", "と" },
    { "na", "な" }, { "ni", "に" }, { "nu", "ぬ" }, { "ne", "ね" }, { "no", "の" },
    { "ha", "は" }, { "hi", "ひ" }, { "fu", "ふ" }, { "he", "へ" }, { "ho", "ほ" },
    { "ma", "ま" }, { "mi", "み" }, { "mu", "む" }, { "me", "め" }, { "mo", "も" },
    { "ya", "や" }, { "yu", "ゆ" }, { "yo", "よ" },
    { "ra", "ら" }, { "ri", "り" }, { "ru", "る" }, { "re", "れ" }, { "ro", "ろ" },
    { "wa", "わ" }, { "wo", "を" },
    { "n", "ん" },
    // Add more mappings as needed
};
```

Figure 5.1.3.2 Code 3.2 (HiraganaChecker)

In the HiraganaChecker script a Dictionary is used to pair all Hiragana characters with their corresponding Romaji Characters. The paired characters are then assigned with other Dictionaries to check the correctAttempts, inCorrectAttempts and giveUpAttempts of every character for performance evaluation later.

```
public Dictionary<string, int> romajiToTileIndex; // Map romaji to tile indices
public Dictionary<string, int> progressionCounters = new Dictionary<string, int>(); // Correct answer counters
```

Figure 5.1.3.3 Code 3.3 (ProgressionUI)

The ProgressionUI script uses the dictionary in Code 3.2 at the start of the game to pair the romajiToIndex to match each Hiragana to the tiles on the progression panel that indicates the correct progression with percentage of segment of the tile displayed. It also pairs the progressionConter to track if each character has reached its max score which is a condition to level up.

```
private readonly Dictionary<string, string> level1Characters = new Dictionary<string, string>
{
    { "a", "あ" }, { "i", "い" }, { "u", "う" }, { "e", "え" }, { "o", "お" },
};

private readonly Dictionary<string, string> level2Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level3Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level4Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level5Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level6Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level7Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level8Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level9Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level10Characters = new Dictionary<string, string>...;

private readonly Dictionary<string, string> level11Characters = new Dictionary<string, string>...;
```

Figure 5.1.3.4 Code 3.4 (LevelManager)

Later, few Dictionaries are used to initialize the characters to be used on each level. These characters will then be assigned to the romajiToHiragana dictionary during the start of the level to assign different characters to be set in different levels.

5.1.4 Language Changing

- **Download language traineddata of your choice form "https://github.com/tesseract-ocr/tessdata"**
- **Drop the file into "...\Assets\StreamingAssets\tessdata" in the unity editor.**
- **Switch to your language file name as Code 3.1 in the TesseractDriver script.**
- **Edit your characters of each level in the iraganaChecker and LevelManager script as shown in Code 3.2, 3.3 and 3.4.**
- **Change the position of tiles and character table image of the progression panel.**
- **Save and rebuild the game.**

5.1.5 Question Spawning

```
public void SetRandomRomaji()
{
    // Select a random romaji key from the dictionary and set it as the current romaji
    List<string> keys = new List<string>(romajiToHiragana.Keys);
    currentRomaji = keys[Random.Range(0, keys.Count)];
```

Figure 5.1.4.1 Code 4.1

The SetRandomRomaji method in HiraganaChecker selects a random romaji key from the romajiToHiragana dictionary and assigns it to currentRomaji. It first creates a list of all the keys in the dictionary, then uses Random.Range to pick a random key. The random romaji is later used in the assignment of questions in the QuestionSpawner.

```
public void MoveToRandomStudent()
{
    // Select a random student from the list and move the QuestionSpawner to their position + offset
    currentStudent = students[Random.Range(0, students.Count)];
    Vector3 offsetPosition = currentStudent.position + new Vector3(offsetX, offsetY, offsetZ);

    // Move the QuestionSpawner to the new position with offsets
    transform.position = offsetPosition;

    currentStudentAnimator = currentStudent.GetComponent<Animator>();
    if (currentStudentAnimator != null)
    {
        currentStudentAnimator.SetBool("asking", true);  // Set asking to true for the new student
    }

    HideQuestion();
}
```

Figure 5.1.4.2 Code 4.2

```
public IEnumerator DisplayQuestionWithDelay()
{
    // Wait for some time for animation to play first
    yield return new WaitForSeconds(delay);

    // Display the romaji question and hiragana answers
    hiraganaChecker.SetRandomRomaji();
    DisplayRomajiQuestion();
}
```

Figure 5.1.4.3 Code 4.3

The QuestionSpawnerScripts makes the question spawner move to a random student at the start of the game and triggers its “asking” animation. It also hides the question bubbles as this method is also used when the player answers the question correctly and makes the question spawner move to another student. Hiding the question and respawning it with delay and popping animation matches the timing of the question spawning when the student raises its hand and enhances its visual experience.

```csharp
private void DisplayRomajiQuestion()
{
    AudioManager.instance.Play(popUpAudioName);

    // Show romaji text
    romajiText.enabled = true;
    bubbles[0].SetActive(true);
    anim.Play();

    // Get the current romaji from the HiraganaChecker
    currentRomaji = hiraganaChecker.CurrentRomaji;

    // Display the romaji in the UI
    romajiText.text = currentRomaji;

    // Set up hiragana answers, including one correct answer and four random incorrect ones
    List<string> answers = new List<string> { hiraganaChecker.romajiToHiragana[currentRomaji] }; // Correct answer

    // Add random incorrect answers, ensuring no duplicates
    while (answers.Count < 5)
    {
        string randomAnswer = hiraganaChecker.romajiToHiragana[hiraganaChecker.romajiToHiragana.Keys.
            ElementAt(Random.Range(0, hiraganaChecker.romajiToHiragana.Count))];
        if (!answers.Contains(randomAnswer))
        {
            answers.Add(randomAnswer);
        }
    }

    // Shuffle answers to randomize the order
    answers = answers.OrderBy(x => Random.value).ToList();

    // Display each hiragana answer in the corresponding Text UI
    for (int i = 0; i < hiraganaTexts.Count; i++)
    {
        hiraganaTexts[i].text = answers[i];

        // Change the color of the correct answer based on the counter
        if (answers[i] == hiraganaChecker.romajiToHiragana[currentRomaji])
        {
            if (progressionUI.progressionCounters[currentRomaji] >= hideThreshold)
            {
                // Hide all hiragana answer texts
                foreach (var text in hiraganaTexts)
                {
                    text.text = "";
                }
                return;
            }
            else if (progressionUI.progressionCounters[currentRomaji] < colorThreshold)
            {
                hiraganaTexts[i].color = Color.blue; // Display in blue if below correctThreshold
            }
            else
            {
                hiraganaTexts[i].color = Color.black; // Display in black if above correctThreshold but below hideThreshold
            }
        }
        else
        {
            hiraganaTexts[i].color = Color.black; // Default color for incorrect answers
        }
    }
    ShowQuestion();
}
```

Figure 5.1.4.4 Code 4.4

The DisplayRomajiQuestion method is responsible for displaying a romaji-based question to the player. The current romaji character retrieves a random Romaji from the HiraganaChecker and displays it in the spawner. The method then generates a list of answers, with the correct hiragana corresponding to the current romaji and four randomly selected incorrect hiragana characters. These answers are shuffled and displayed in bubbles in random order. If the character’s progression is below the colorTreshold it will appear in blue as a starter hint or else it will be in black color as other characters as normal hint. When the correctProgression is greater or equal to the hideThreshold all hints will be hidden and no bubble will be shown.

5.1.6 Character Checking

```
public void CheckDrawing()
{
    if (playerDrawing == null) return;

    Texture2D texture = ConvertRenderTextureToTexture2D(playerDrawing);

    // Recognize the drawn character
    recognizedCharacter = _tesseractDriver.Recognize(texture);
    expectedCharacter = romajiToHiragana[currentRomaji];

    read.text = recognizedCharacter;
}
```

Figure 4.1.5.1 Code 5.1

The CheckDrawing method in the HiraganaChecker script checks if there is a drawing to evaluate when the player finishes drawing and after the player clicks the check button. It then converts the drawn texture (playerDrawing) from a RenderTexture to a Texture2D. After that, the method uses the Recognize method from TesseractDriver script to recognize the drawn character and assigns it to recognizedCharacter. It also retrieves the expected character from a mapping of romaji to hiragana. Finally, the recognized character is displayed on the read.text UI element for feedback for the player to check if their drawn character is recognized.

```
// Compare and display result
if (NormalizeString(recognizedCharacter) == NormalizeString(expectedCharacter))
{
    AudioManager.instance.Play(correctAudioName);
    score += 100;
    tries = 0;
    correctAttempts[currentRomaji]++;
    scoreDisplay.text = score.ToString();
    resultDisplay.text = "Correct!";
    questionSpawner.OnAnswerChecked(true);
    progressionUI.UpdateTile();
    whiteboard.ClearBoard();
}
```

Figure 4.1.5.2 Code 5.2

```
public void OnAnswerChecked(bool isCorrect)
{
    if (isCorrect)
    {
        if (progressionUI.progressionCounters[currentRomaji] < maxScore)
        {
            progressionUI.progressionCounters[currentRomaji]++;
        }
        levelManager.CheckLevelChanges();
        MoveToRandomStudent();
        StartCoroutine(DisplayQuestionWithDelay());
    }
}
```

Figure 4.1.5.3 Code 5.3

If the checked drawn character is the same as the expected character based on the dictionary when the player clicks the check button, a correct audio will be played. 100 points will be added, the correct attempt and the progression of the character will be increased.

```
else
{
    AudioManager.instance.Play(wrongAudioName);
    tries++;
    incorrectAttempts[currentRomaji]++;

    if (tries >= 3)
    {
        tries = 0;

        if (score > 0)
        {
            score -= 100; // Reduce lesser score if try but wrong

            if (score < 0)
            {
                score = 0;
            }
        }

        if (progressionUI.progressionCounters[currentRomaji] > 0)
        {
            progressionUI.progressionCounters[currentRomaji]--; // Decrement counter if more than 3 try
        }

        progressionUI.UpdateTile();
    }

    scoreDisplay.text = score.ToString();
    resultDisplay.text = (3 - tries) + " Attemps Remaining";
    whiteboard.ClearBoard();
}
```

Figure 4.1.5.4 Code 5.4

If the checked drawn character is not the same as the expected character based on the dictionary clicks the check button, a wrong audio will be played. Player has 3 attempts for answering the question, however every incorrect attempt will also be recorded. If the player tries for more than 3 attempts 100 points will be deducted and the progression of the character will be decreased.

```
public void GiveUp()
{
    hiraganaChecker.ResetTimer();
    if (!hiraganaChecker.giveUpCd)
    {
        StartCoroutine(hiraganaChecker.GiveUpCooldown());
    }

    AudioManager.instance.Play(hiraganaChecker.giveUpAudioName);
    hiraganaChecker.tries = 3;
    hiraganaChecker.giveUpAttempts[currentRomaji]++;
    if (hiraganaChecker.score > 0)
    {
        hiraganaChecker.score -= 200; // Reduce more score if give up

        if (hiraganaChecker.score < 0)
        {
            hiraganaChecker.score = 0;
        }
    }

    if (progressionUI.progressionCounters[currentRomaji] > 0)
    {
        progressionUI.progressionCounters[currentRomaji]--; // Decrement counter if give up
    }

    whiteboard.ClearBoard();
    hiraganaChecker.scoreDisplay.text = hiraganaChecker.score.ToString(); // Update score
    hiraganaChecker.resultDisplay.text = "Lv Too Hard?"; // Display result

    progressionUI.UpdateTile(); // Update progression color

    MoveToRandomStudent(); // Move to the next random student if the answer is correct
    StartCoroutine(DisplayQuestionWithDelay());
}
```

Figure 4.1.5.5 Code 5.5

```
private void Update()
{
    if (isTimerOn)
    {
        if (isTimerRunning)
        {
            currentTime -= Time.deltaTime;

            // Update the timer display
            timerText.text = Mathf.Ceil(currentTime).ToString();

            if (currentTime <= 0)
            {
                isTimerRunning = false;
                questionSpawner.GiveUp();
            }
        }
    }
}
```

Figure 4.1.5.6 Code 5.6

Additionally, if the player presses the give up button or if he exceeds the time limit of the difficulty, 100 points will also be deducted and the progression of the character will also be decreased.

5.1.7 Progression

```
public void UpdateTile()
{
    string currentRomaji = hiraganaChecker.CurrentRomaji;  // Get the current romaji from HiraganaChecker
    int correctCount = progressionCounters[currentRomaji]; // Get correct score form QuestionSpawner
    float correctPercentage = (float)correctCount / questionSpawner.maxScore; // Calculate score percentage

    hiraganaTiles[romajiToTileIndex[currentRomaji]].color = Color.Lerp(startColor, endColor, correctPercentage);
    hiraganaTiles[romajiToTileIndex[currentRomaji]].fillAmount = correctPercentage;
}
```

Figure 4.1.6.1 Code 6.1

The UpdateTile method in ProgressonUI script updates the tile color and display area based on the progression percentage of the current romaji. This method is used every time in methods that change the score to ensure immediate feedback on the current progression.

```
public void CheckLevelChanges()
{
    allReachedMaxScore = true;

    foreach (var pair in hiraganaChecker.romajiToHiragana)
    {
        // Check if the current progression counter matches maxScore
        if (progressionUI.progressionCounters[pair.Key] < questionSpawner.maxScore)
        {
            allReachedMaxScore = false; // Set to false if any pair hasn't reached maxScore
            break; // Exit early since we found one that doesn't match
        }
    }

    if (allReachedMaxScore)
    {
        LevelUp();
    }
}
```

Figure 4.1.6.2 Code 6.2

The game runs with a LevelManager script where it checks if all progression of characters on the current level reaches its max progression counter. This checking is performed if the player answers the question correctly as in code 5.3. When all the characters on the level reach the max score, the level increases.

```
public void SetLevel(int level)
{
    outsideSchool.transform.position = new Vector3(transform.position.x, originalOutsideHeight.y - level * 2, transform.position.z);
    hiraganaChecker.romajiToHiragana.Clear();
    currentLevelDisplay.text = "Level " + level.ToString();
    materialColor.ChangeMaterialByLevel(level);
    _tesseractDriver = new TesseractDriver();
    switch (level)
    {
        case 1:
            _tesseractDriver.whitelist = "あいうえお";
            foreach (var pair in level1Characters)
            {
                hiraganaChecker.romajiToHiragana.Add(pair.Key, pair.Value);
            }
            break;
```

Figure 4.1.6.3 Code 6.3

Switching level reinitializes the dictionary in romajiToHiragana with the level’s dictionary. It also resets the whitelist to focus on the characters of the level.

```
public void CalculateLevelResults()
{
    // Initialize counters
    int totalCorrect = 0;
    int totalIncorrect = 0;
    int totalGiveUp = 0;

    // Sum the results for only the keys present in romajiToHiragana
    foreach (var romaji in hiraganaChecker.romajiToHiragana.Keys)
    {
        totalCorrect += hiraganaChecker.correctAttempts[romaji];
        totalIncorrect += hiraganaChecker.incorrectAttempts[romaji];
        totalGiveUp += hiraganaChecker.giveUpAttempts[romaji];
    }

    // Total attempts
    float totalAttempts = totalCorrect + totalIncorrect + totalGiveUp;

    // Calculate percentages
    float correctPercent = totalCorrect / totalAttempts;
    float incorrectPercent = totalIncorrect / totalAttempts;
    float giveUpPercent = totalGiveUp / totalAttempts;

    // Update the pie chart
    correctImage.fillAmount = correctPercent; // Green segment
    incorrectImage.fillAmount = correctPercent + incorrectPercent; // Red segment
    giveUpImage.fillAmount = 1f; // Yellow segment (fills the remaining space)

    // Display results with two decimal places
    totalCorrectTMPDisplay.text = (correctPercent * 100).ToString("F2") + "%";
    totalIncorrectTMPDisplay.text = (incorrectPercent * 100).ToString("F2") + "%";
    totalGiveUpTMPDisplay.text = (giveUpPercent * 100).ToString("F2") + "%";
    totalQuestionAnswered.text = totalAttempts.ToString();

    // Avoid division by zero
    if (totalAttempts == 0)
    {
        totalCorrectTMPDisplay.text = "0%";
        totalIncorrectTMPDisplay.text = "0%";
        totalGiveUpTMPDisplay.text = "0%";
        totalQuestionAnswered.text = "0";
        return;
    }
}
```

Figure 4.1.6.4 Code 6.4

At the end of each level, a performance of the level will be displayed, showing the total correct, incorrect and give up percentage of the level.

```
public void FinalDisplay()
{
    progressionPanel.SetActive(true);

    foreach (var tile in hiraganaTiles)
    {
        tile.color = startColor;

        if (tile is Image image)
        {
            // Set the image type to Filled Radial360
            image.type = Image.Type.Filled;
            image.fillMethod = Image.FillMethod.Radial360;
            image.fillAmount = 0f;
        }
    }

    // Loop through each romaji in progressionCounters
    foreach (var pair in progressionCounters)
    {
        string currentRomaji = pair.Key;  // Get the current romaji
        int correctCount = pair.Value;    // Get the correct score from progressionCounters
        int incorrectCount = hiraganaChecker.incorrectAttempts[currentRomaji];
        int giveUpCount = hiraganaChecker.giveUpAttempts[currentRomaji];

        int totalAttempts = correctCount + incorrectCount + giveUpCount;
        float correctPercentage = (float)correctCount / totalAttempts;  // Calculate score percentage


        if (totalAttempts > 0)
        {
            // Update the tile's color and fill amount for each romaji
            hiraganaTiles[romajiToTileIndex[currentRomaji]].color = Color.Lerp(startColor, endColor, correctPercentage);
            hiraganaTiles[romajiToTileIndex[currentRomaji]].fillAmount = correctPercentage;
        }
        else
        {
            // Optionally reset the tile if no attempts were made
            hiraganaTiles[romajiToTileIndex[currentRomaji]].color = startColor;
            hiraganaTiles[romajiToTileIndex[currentRomaji]].fillAmount = 0;
        }
    }
}
```

Figure 4.1.6.5 Code 6.5

At the end of the game when all levels are finished, an additional total performance of the correct percentage of each Hiragana character will be displayed using the progression panel.

5.2 Testing Strategy / Approaches

5.2.1 Unit testing

- **Player Drawing Recognition:** Test if Tesseract OCR correctly recognizes characters drawn by the player.
- **Performance and Frame Rate:** Use Unity Profiler to test frame rates when the player is drawing and checking the Hiragana character.
- **Question Spawning:** Verify the Question Spawner correctly selects a random student and displays the romaji and corresponding hiragana answers.

5.2.2 Beta testing

- **Ease of Use in VR:** Test if players can intuitively draw characters and interact with the UI using VR controllers.
- **Learning Effectiveness:** Evaluate whether players feel their ability to write and recognize hiragana has improved.
- **Game Balance:** Test whether the progression system is too fast or slow for typical users.

5.2.3 White box testing

- **OCR Logic:** Test the CheckDrawing method to ensure it correctly compares the recognized character with the expected character.
- **Progression System:** Verify progression counters increment correctly and trigger level-up conditions as expected.
- **Answer Shuffling:** Ensure the DisplayRomajiQuestion method always includes one correct answer and no duplicates among incorrect answers.

5.2.4 Black box testing

- **Gameplay Flow:** Test if the game progresses correctly when a player answers questions, updating scores and resetting for the next round.
- **UI Feedback:** Ensure feedback mechanisms like audio cues, vibrations, and color changes work as expected.
- **Edge Cases:** Test scenarios like leaving the whiteboard empty, drawing invalid characters, or selecting invalid answers.

5.3 Test Plan

5.3.1 Test Objectives

- Ensure all core functionalities of "Hiragana Sensei" operate as expected.
- Verify that the educational goals of teaching hiragana are met through gameplay.
- Validate system performance, usability, and user experience in a VR environment.

5.3.2 Scope

This test plan covers the following components:

- Writing Mechanic
- Tesseract OCR Integration
- Hiragana Checker
- Question Spawner
- Progression Tracking and Display
- Performance Display
- Draw Effects

5.3.3 Test Items

- VR interaction for drawing and UI interaction.
- Optical Character Recognition (OCR) for player handwriting.
- Randomization and correctness of questions and answers.
- Progression system for tracking player's learning.

5.3.4 Features to be Tested

1. **Writing Mechanic**
    - Accuracy of drawing detection.
    - Responsiveness of haptic feedback during drawing.

2. **Tesseract OCR**
    - Recognition of handwritten hiragana.
    - Accuracy of OCR in differentiating correct and incorrect characters.

3. **Hiragana Checker**
    - Validation of drawn characters against correct answers.
    - Proper feedback display for correct and incorrect responses.

4. **Question Spawner**
    - Random selection of students and accurate display of romaji and answers.
    - Correct placement and visibility of Text UI elements.

5. **Progression Tracking and Display**
    - Correct tracking of player progress for each character.
    - Proper update and display of progression tiles.

6. **Performance Display**
    - Accuracy in displaying game performance, including scores and progress.

5.3.5 Test Approach

Tests will be conducted using both manual and automated testing methods in the following environments:

- VR headset and controllers.
- Desktop testing for non-VR functionalities.

5.4 Test Case

5.4.1 Test Case 1 (VR)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_VR	**Test Designed by:** Lai Kah Hoe
Test Priority (Low/Medium/High): Med	**Test Designed date:** 30 Nov 2024
Module Name: Menu, Tutorial, Classroom	**Test Executed by:** Lee Seng Wai
Test Title: Test VR	**Test Execution date:** 7 Dec 2024
Description: Test UI and Buttons in the game	

Pre-conditions: The player has very little experience with VR devices.
Dependencies: User input.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Does not feel dizzy while playing the game.	VR motion sickness.	The player will not feel dizzy within 15 minutes.	The player does not feel dizzy within 15 minutes.	Pass	
2	Identify UI elements.	Game UI	The player is able to identify and read the game UI clearly.	The player is able to identify and read the game UI clearly.	Pass	
3	Look around with the continuous turn locomotion system.	Game turn controls	The player is able to look around by turning with the continuous turn locomotion system.	The player is able to look around by turning with the continuous turn locomotion system.	Pass	
4	Look around with the snap turn locomotion system.	Game turn controls	The player is able to look around by turning with the snap turn locomotion system.	The player is able to look around by turning with the snap turn locomotion system.	Pass	
5	Look around by moving the head only.	VR view rotation	The player is able to look around by turning his head only.	The player is able to look around by turning his head only.	Pass	
6	Move around	Game input controls	The player is able to move around by using the controller.	The player is able to move around by using the controller.	Pass	

Post-conditions:
Player ables to perform basic drawings and is able to clear the whiteboard. Player is able to receive feedback on his actions.

5.4.2 Test Case 2 (UI)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_UI

Test Designed by: Lai Kah Hoe

Test Priority (Low/Medium/High): Low

Test Designed date: 30 Nov 2024

Module Name: Menu, Tutorial, Classroom

Test Executed by: Aloysius Cheong Ken Wai

Test Title: Test UI and Buttons

Test Execution date: 7 Dec 2024

Description: Test UI and Buttons in the game

Pre-conditions: The player has never played this game before.
Dependencies: User input.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Notice button hovered..	Game buttons, button hovered audio	Players are able to notice the button hovered with color change on button and line renderer as well as hearing hover enter audio.	Players are able to notice the button hovered with color change on button and line renderer as well as hearing hover enter audio.	Pass	
2	Notice button selected.	Game buttons, button clicked audio	Players are able to notice the button clicked with color change on button as well as hearing button clicked audio.	Players are able to notice the button clicked with color change on button as well as hearing button clicked audio.	Pass	
3	Understand UI and buttons in the main menu.	Main menu buttons	Player ables to understand the main menu UI and buttons without telling.	Player ables to understand the main menu UI and buttons without telling.	Pass	
4	Understand UI text and arrows guide in tutorial.	Tutorial Ui and buttons	Player ables to understand the UI text and arrows guide in the tutorial.	Player ables to understand the UI text and arrows guide in the tutorial.	Pass	
5	Understand UI and button functions at game level.	Game level buttons	Player ables to understand UI and button functions at game level.	Player ables to understand UI and button functions at game level.	Pass	
6	Able to select buttons easily.	Buttons in game	Player ables to select buttons easily throughout the game.	Player ables to select buttons easily throughout the game.	Pass	

Post-conditions:
Player ables to perform basic drawings and is able to clear the whiteboard. Player is able to receive feedback on his actions.

5.4.3 Test Case 3 (Drawing)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Drawing	**Test Designed by:** Lai Kah Hoe
Test Priority (Low/Medium/High): High	**Test Designed date:** 30 Nov 2024
Module Name: Classroom	**Test Executed by:** Koo Weng Shen
Test Title: Drawing Test	**Test Execution date:** 7 Dec 2024
Description: Test drawing on whiteboard	

Pre-conditions: Player does not have experience drawing with VR controllers.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Hold the grip button to draw on the whiteboard.	Drawing mechanism	Player ables to draw something on the whiteboard by holding the grip button.	Player ables to draw something on the whiteboard by holding the grip button.	Pass	
2	Feel the vibration of the controller when drawing.	Controller vibration	Player can tell that his drawing input was accepted with haptic feedback.	Player can tell that his drawing input was accepted with haptic feedback.	Pass	
3	Select the duster to clear the board.	Whiteboard clearing	Whiteboard is clear to default color when the duster is selected.	Whiteboard is clear to default color when the duster is selected.	Pass	
4	Draw straight lines on the whiteboard 10 times.	Player input	10 not perfect straight lines will be drawn on the whiteboard.	Lines drawn are not fully straight due to lack of hand supporting area and the lines are not fully smooth.	Pass	The results are still acceptable.
5	Draw squares/rectangles on the whiteboard 10 times.	Player input	10 not perfect squares/rectangles will be drawn on the whiteboard.	Squares/Rectangles drawn are not fully perfect due to lack of hand supporting area and the lines are not fully smooth.	Pass	The results are still acceptable.
6	Draw circles on the whiteboard 10 times.	Player input	10 not perfect circles will be drawn on the whiteboard.	Circles drawn are not fully perfect due to lack of hand supporting area and the lines are not fully smooth.	Pass	The results are still acceptable.

Post-conditions:
Player ables to perform basic drawings and is able to clear the whiteboard. Player is able to receive feedback on his actions.

5.4.4 Test Case 4 (Tutorial)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Lv0

Test Designed by: Lai Kah Hoe

Test Priority (Low/Medium/High): Med

Test Designed date: 9 Dec 2024

Module Name: Tutorial

Test Executed by: Ng Han Wei

Test Title: Tutorial Level Testing

Test Execution date: 16 Dec 2024

Description: Test the tutorial level

Pre-conditions: The player knows how to write Hiragana characters in real life.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Read and proceed with tutorial instructions.	Tutorial guide text	Player understands the tutorial guide text.	Player understands the tutorial guide text.	Pass	
2	Identify what Hiragana character to write	Question and hints spawned	Player ables to identify Hiragana to be written based on the Romaji questions and the hint system.	Player able to identify Hiragana to write before noticing hints as he already knows how to write Hiragana characters.	Pass	Passed with pre-condition
3	Write character as requested in the tutorial level.	Player's drawing input.	Player's drawing is able to be recognized by the game.	Write character as requested in the tutorial level.	Fail	
4	Repeat step 3	Player's drawing input.	Character will be easier to recognize as player draw better	50% of characters drawn are recognizable when the player draws the character very nicely.	Pass	
5	Recognizing inputs	Input buttons	Player remembers the draw and select button.	Player remembers the draw and select button.	Pass	
6	End tutorial	End tutorial button	Player ends the tutorial by pressing the skip tutorial button.	Player ends the tutorial by pressing the skip tutorial button.	Pass	

Post-conditions:
Player passes and understands the tutorial level. The game is able to recognize characters written by the player.

5.4.5 Test Case 5 (Lv1)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Lv1	**Test Designed by:** Lai Kah Hoe
Test Priority (Low/Medium/High): High	**Test Designed date:** 30 Nov 2024
Module Name: Classroom	**Test Executed by:** Cheong Khe Yang
Test Title: Level 1 Test	**Test Execution date:** 7 Dec 2024
Description: Test level 1	

Pre-conditions: The player knows how to write Hiragana characters in real life.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Test drawing character "あ" 10 times.	Player input	5/10 of the characters "あ" will be recognizable by the OCR.	8/10 of the character "あ" were recognized by the OCR.	Pass	
2	Test drawing character "い" 10 times.	Player input	5/10 of the characters "い" will be recognizable by the OCR.	5/10 of the characters "い" were recognized by the OCR.	Pass	
3	Test drawing character "う" 10 times.	Player input	5/10 of the characters "う" will be recognizable by the OCR..	8/10 of the characters "う" were recognized by the OCR.	Pass	
4	Test drawing character "え" 10 times.	Player input	5/10 of the characters "え" will be recognizable by the OCR..	7/10 of the characters "え" were recognized by the OCR.	Pass	
5	Test drawing character "お" 10 times.	Player input	5/10 of the characters "お" will be recognizable by the OCR.	9/10 of the characters "お" were recognized by the OCR.	Pass	
6	Check progression	Progression UI	Player understands how to check progression UI.	Player understands how to check progression UI.	Pass	
7	Complete level 1	Level manager	Player able to complete level 1.	Player able to complete level 1.	Pass	
8	Check level performance	Level Manager	Player can view his performance at the end of the level.	Player can view his performance at the end of the level.	Pass	

Post-conditions:
The OCR system is able to recognize most Hiragana characters in level 1. Player complete level 1.

5.4.6 Test Case 6 (Lv2)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Lv2	**Test Designed by:** Lai Kah Hoe
Test Priority (Low/Medium/High): High	**Test Designed date:** 30 Nov 2024
Module Name: Classroom	**Test Executed by:** Chan Chi Keat
Test Title: Level 2 Test	**Test Execution date:** 7 Dec 2024
Description: Test level 2	

Pre-conditions: The player knows how to write Hiragana characters in real life.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Test drawing character "か" 10 times.	Player input	5/10 of the characters "か" will be recognizable by the OCR.	5/10 of the characters "か" were recognized by the OCR.	Pass	
2	Test drawing character "き" 10 times.	Player input	5/10 of the characters "き" will be recognizable by the OCR.	5/10 of the characters "き" were recognized by the OCR.	Pass	
3	Test drawing character "く" 10 times.	Player input	5/10 of the characters "く" will be recognizable by the OCR.	9/10 of the characters "く" were recognized by the OCR.	Pass	
4	Test drawing character "け" 10 times.	Player input	5/10 of the characters "け" will be recognizable by the OCR.	10/10 of the characters "け" were recognized by the OCR.	Pass	
5	Test drawing character "お" 10 times.	Player input	5/10 of the characters "こ" will be recognizable by the OCR.	5/10 of the characters "こ" were recognized by the OCR.	Pass	
6	Check progression	Progression UI	Player understands how to check progression UI.	Player understands how to check progression UI.	Pass	
7	Complete level 2	Level manager	Player able to complete level 2.	Player able to complete level 2.	Pass	
8	Check level performance	Level Manager	Player can view his performance at the end of the level.	Player can view his performance at the end of the level.	Pass	

Post-conditions:
The OCR system is able to recognize most Hiragana characters in level 2.

5.4.7 Test Case 7 (Lv3)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Lv3	**Test Designed by:** Lai Kah Hoe
Test Priority (Low/Medium/High): High	**Test Designed date:** 30 Nov 2024
Module Name: Classroom	**Test Executed by:** Khoo Qi Rui
Test Title: Level 3 Test	**Test Execution date:** 7 Dec 2024
Description: Test level 3	

Pre-conditions: The player knows how to write Hiragana characters in real life.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Test drawing character “さ” 10 times.	Player input	5/10 of the characters “さ” will be recognizable by the OCR.	7/10 of the characters “さ” were recognized by the OCR.	Pass	
2	Test drawing character “し” 10 times.	Player input	5/10 of the characters “し” will be recognizable by the OCR.	10/10 of the characters “し” were recognized by the OCR.	Pass	
3	Test drawing character “す” 10 times.	Player input	5/10 of the characters “す” will be recognizable by the OCR.	9/10 of the characters “す” were recognized by the OCR.	Pass	
4	Test drawing character “せ” 10 times.	Player input	5/10 of the characters “せ” will be recognizable by the OCR.	7/10 of the characters “せ” were recognized by the OCR.	Pass	
5	Test drawing character “そ” 10 times.	Player input	5/10 of the characters “そ” will be recognizable by the OCR.	5/10 of the characters “そ” were recognized by the OCR.	Pass	
6	Check progression	Progression UI	Player understands how to check progression UI.	Player understands how to check progression UI.	Pass	
7	Complete level 3	Level manager	Player able to complete level 3.	Player able to complete level 3.	Pass	
8	Check level performance	Level Manager	Player can view his performance at the end of the level.	Player can view his performance at the end of the level.	Pass	

Post-conditions:
The OCR system is able to recognize most Hiragana characters in level 3.

5.4.8 Test Case 8 (Lv Extra)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_LvExtra

Test Designed by: Lai Kah Hoe

Test Priority (Low/Medium/High): Med

Test Designed date: 9 Dec 2024

Module Name: Tutorial

Test Executed by: Chan Chee Yung

Test Title: Test Level Extra

Test Execution date: 16 Dec 2024

Description: Test answering with no hint bubbles.

Pre-conditions: The player knows how to write Hiragana characters in real life.
Dependencies: User input tracking.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Identify what character to write.	Question and hints spawned.	Player ables to identify Hiragana to be written based on the romaji questions only (without hints).	Player ables to identify Hiragana to be written based on the romaji questions only (without hints).	Pass	Passed with pre-condition
2	Write Hiragana characters corresponding to romaji.	Player's drawing input.	Player's is able to write the correct Hiragana.	Hiragana is written correctly but has a low recognition rate.	Pass	
3	Repeat step 2 with better handwriting.	Player's drawing input.	Characters will be easier to recognize as the player draws better.	Hiragana is written correctly and has a higher recognition rate.	Pass	
4	Complete the level.	Player drawing input.	Player ables to complete the level.	Player is too lazy to finish the level as level 3 requires the player to answer correctly for 20 times on each character. (level too long).	Fail	

Post-conditions:
Player passes the level. The game is able to recognize characters written by the player.

5.4.9 Test Case 9 (Draw Effects)

Project Name: Hiragana Sensei

Test Case Template

Test Case ID: Test_Lv1

Test Designed by: Lai Kah Hoe

Test Priority (Low/Medium/High): Low

Test Designed date: 30 Nov 2024

Module Name: Shop, Classroom

Test Executed by: Lai Johnson

Test Title: Test Draw Effects

Test Execution date: 7 Dec 2024

Description: test different color and drawing's particle effects

Pre-conditions: The player has never played this game before.
Dependencies: User input.

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Open shop in the main menu.	Main menu, Shop	The player is able to navigate to the shop.	The player is able to navigate to the shop.	Pass	
2	Buy a new draw color.	Draw color	The player is able to buy a new draw color.	The player is able to buy a new draw color.	Pass	
3	Select different draw colors.	Draw color	The player is able to select a different draw color.	The player is able to select a different draw color.	Pass	
4	Buy a new drawing's particle effect.	Draw particle effect	The player is able to buy a new draw particle effect.	The player is able to buy a new draw particle effect.	Pass	
5	Select a new drawing's particle effect.	Draw particle effect	the player is able to select a different draw particle effect.	the player is able to select a different draw particle effect.	Pass	
6	Draw on a whiteboard with the new color and particle effect.	Draw color, Draw particle effect	Drawing color and particle effects are working and same as selected by the player.	Drawing color and particle effects are working and same as selected by the player.	Pass	
7	Repeat step 2-6	Draw color, Draw particle effect	The player is able to perform step 2-6 with different variations of effects.	The player is able to perform step 2-6 with different variations of effects.	Pass	

Post-conditions:
Player ables to test all drawing effects. All drawing effects are working as expected.

5.5 Chapter Summary and Evaluation

This chapter presented the implementation and testing of Hiragana Sensei. The implementation section covered the core mechanics of the game, including writing mechanics, the Tesseract OCR wrapper and driver, Hiragana setup, question spawning, character checking, progression tracking, and performance display. Detailed code explanations demonstrated how each component was developed to provide an engaging and educational gameplay experience. The flexible code structure allows easy customization, making it adaptable for different languages or educational objectives. The testing strategy encompassed Unity-based testing, beta testing, and black-box and white-box testing, ensuring a comprehensive evaluation of the game's functionality and user experience. The test plan and test cases highlighted critical areas such as VR interactions, UI functionality, and gameplay across all levels. Key findings included a test case revealing that the OCR system achieved a recognition accuracy of only 50% for specific Hiragana characters, suggesting potential areas for optimization. The implementation demonstrated a robust integration of game mechanics and educational elements, showcasing a strong emphasis on user interaction and learning outcomes. The use of Tesseract OCR for handwriting recognition provided a unique feature but also revealed limitations in character recognition accuracy for certain Hiragana, warranting further refinement. The game's structure was noted for its flexibility, allowing seamless adaptation to additional languages or expanded features in the future. The testing phase confirmed the stability and functionality of core mechanics, while also identifying improvement areas, particularly in handwriting recognition and user feedback. Overall, the project successfully balanced technical innovation with educational value, laying a strong foundation for further enhancements.

Chapter 6

Discussions and Conclusion

6 Discussions and Conclusion

Each student is required to make an *evaluation of the project* he/she has embarked on. The project evaluation may include the following sections.

IMPORTANT NOTE TO STUDENTS: In this chapter, for problems related to code, hardware, internet connection, etc (where applicable):
List the technical problems faced and state how they were resolved
List the unsolved technical problems for future enhancement
List the achieved objectives/modules
List the incomplete parts for future enhancement - this is regardless the parts listed in the pre-determined scope. Suggestions for future improvement

6.1 Summary

Hiragana Sensei is a virtual reality educational game created to enhance the learning of Japanese Hiragana characters through an immersive and interactive experience. The player assumes the role of a teacher in a VR classroom, where students raise their hands to ask for the Hiragana equivalent of displayed Romaji. The game's core mechanic requires players to draw the correct character, which is then evaluated using Tesseract OCR for accuracy. The gameplay is further enriched by a dynamic question spawning system that randomly selects a student and generates a set of possible answers, including the correct one and distractors. This system ensures a varied and challenging experience with each playthrough.

The game is built using Unity, integrating advanced features like XR Interaction Toolkit for seamless VR interaction. A robust writing system captures the player's input and processes it for recognition, while progression tracking allows the game to monitor the player's performance and adjust difficulty dynamically. Players are guided through multiple levels, starting with basic tutorials and progressing to advanced challenges, including stages with no hints and added visual effects for a stimulating experience.

Hiragana Sensei also emphasizes replayability and adaptability. Its code structure is designed to be flexible, enabling easy customization for different educational content or languages beyond Hiragana. By combining the latest VR technology with thoughtful game design, Hiragana Sensei provides an engaging, personalized, and effective way to learn Japanese characters, making it a valuable tool for both beginners and enthusiasts.

6.2 Achievement

Effective Handwriting Recognition with OCR

One of the key achievements of Hiragana Sensei is the integration of Tesseract OCR to recognize the player's handwriting. This enables the game to evaluate the player's ability to write Japanese Hiragana characters in real-time. The system checks if the character is written correctly and provides immediate feedback, promoting a learning-by-doing approach. This feature makes the game stand out from traditional language-learning applications.

Immersive VR Learning Experience

The game achieves a fully immersive learning environment using Virtual Reality (VR) technology. Players interact with a classroom setting where they must teach and respond to students, providing an authentic and engaging experience. By leveraging VR, Hiragana Sensei goes beyond conventional language learning tools, offering a more interactive and visual method for mastering Japanese Hiragana.

Unique Position in Market

One of the key achievements of Hiragana Sensei is its unique position in the educational game market. Unlike many other educational games, this game focuses on teaching the Japanese Hiragana script in a VR environment, offering an immersive experience that combines learning with interaction. The use of VR to teach writing is a novel approach, making it stand out from traditional educational tools that primarily rely on 2D screens or physical books. This VR-based learning platform provides a more engaging and dynamic way for players to learn, interact, and practice writing, creating a distinctive niche in the educational gaming industry.

Customizable Language Support

Hiragana Sensei is designed to be highly customizable. The flexible code structure allows the addition of new languages, making it easier to adapt the game for other learning needs. By providing a simple framework to integrate new character sets and mappings, the game can potentially be expanded to teach various languages in the future, broadening its educational impact.

6.3 Contributions

Advancing Educational VR Games

Hiragana Sensei contributes to the VR industry by demonstrating the potential of immersive environments for educational purposes. Unlike traditional VR games, this project focuses on using the technology to teach a new skill (Japanese Hiragana) in a highly interactive way. By incorporating real-time handwriting recognition, it offers a compelling example of how VR can be used to enhance educational tools and foster deeper learning.

Innovative Approach to Language Learning

The project significantly contributes to the field of educational games by blending gamification with language learning. By using both writing mechanics and real-time feedback, Hiragana Sensei innovates the traditional language-learning model. The game creates an engaging and enjoyable way to learn, making it easier for users to retain knowledge through active participation, rather than passive memorization.

Introducing Adaptive Difficulty in Educational Games

Hiragana Sensei introduces a unique approach to adaptive difficulty, where the game dynamically adjusts based on the player's progress. This helps to keep players engaged while maintaining an appropriate level of challenge. The ability to adjust difficulty ensures that players are always motivated to improve, making learning more effective. This concept has great potential for further use in educational games across various subjects.

Demonstrating the Potential of OCR in Education

The use of Tesseract OCR to recognize handwritten characters in a VR environment contributes to the broader field of educational technology. It highlights how OCR can be integrated into immersive games to provide real-time feedback, which could be adapted to various educational contexts. This application of OCR technology opens new possibilities for interactive learning experiences.

Enhancing the Accessibility of Language Learning

By making the game highly customizable and adaptable to different languages, Hiragana Sensei contributes to the accessibility of language learning. It can easily be expanded to teach not just Japanese, but also other languages with different alphabets. This flexibility can provide opportunities for learners worldwide to use the game to study various languages, broadening its impact and appeal.

6.4 Limitations

Time Constraints Due to Short Semester Duration

One of the primary limitations faced during the development of Hiragana Sensei was the limited time frame imposed by the short semester. With only a few weeks available to complete the project, there were constraints on the amount of features and polish that could be added. This affected the overall scope of the game, limiting some features that could have enhanced the learning experience further, such as more diverse levels.

Limited Access to VR Devices

Another significant limitation was the lack of access to VR devices at home. As VR equipment is not readily available, the development process was heavily reliant on borrowing devices from the school and working in the lab. This meant that testing and iteration had to be done within specific timeframes, which at times limited the ability to test the game thoroughly or make last-minute changes based on user feedback.

Insufficient Asset Funding

Due to the lack of funding for assets, the visual quality of the game's assets was not as high as desired. The textures, character models, and environments were created with limited resources, which impacted the overall aesthetic appeal of the game. While the game was functional and educational, the lack of polished assets may affect its visual engagement and immersive experience for users, especially in a VR setting where visuals are crucial.

Challenges with Tesseract OCR Training Data

One significant limitation was the inability to prepare a custom-trained OCR model for the game due to time constraints and language differences. While Tesseract OCR is a powerful tool, it requires a well-prepared training dataset to accurately recognize handwritten Hiragana characters. Unfortunately, there was not enough time to gather and preprocess the data needed to train the model for optimal recognition, especially considering the variations in handwriting styles. As a result, the OCR's accuracy was limited, and some characters were recognized with lower precision, which impacted the game's performance in recognizing the player's input correctly.

6.5 Future Improvements

Expanded Levels with Word Recognition

Currently, the game focuses on Hiragana characters, but expanding the levels to include whole words instead of just individual characters could provide a more challenging and enriching experience. Adding more complex levels with word recognition and sentence formation would deepen the learning experience and allow players to progress through a wider range of Japanese language skills. This could also include grammar and sentence structures, broadening the scope of the game to cater to advanced learners.

Enhanced OCR Recognition with Custom Training Data

To improve the accuracy of the OCR system, future improvements could include training a custom model specifically for the game using labeled handwritten data. By collecting handwriting samples from users and curating a dataset, the game's OCR could be fine-tuned to recognize different writing styles more effectively. This would address current limitations in recognizing some characters, making the recognition process more accurate and reliable for players, especially those who might have unique writing styles.

Higher-Quality Assets for Immersion

Another area for improvement is the visual assets in the game. To enhance player immersion, the quality of assets such as textures, models, and environments could be improved. Higher-quality assets would create a more engaging and polished experience, making the classroom and its elements feel more lifelike and interactive. This would help make the VR environment more appealing to players, creating a stronger connection to the in-game world and improving the overall aesthetic of the project.

Cosmetic Customization for Classrooms and Students

Adding customization options for the classroom and the students would be another exciting enhancement. Players could personalize their classroom environment by selecting different themes, furniture, and even customizing the appearance of the students. This would give players a sense of ownership and control over their virtual classroom, adding a layer of creativity and personalization that could enhance the overall enjoyment and replayability of the game.

Player vs. Player (PvP) Mode

One exciting future improvement for Hiragana Sensei would be the addition of a PvP mode, where players can compete against each other to test their Hiragana knowledge. In this mode, players could take turns answering questions or even challenge each other to write the correct characters as quickly as possible. This would introduce a competitive element to the game, encouraging players to practice more and making the game more engaging, especially for multiplayer or online interactions.

Support for Different Language Models

To make Hiragana Sensei more accessible to a broader audience, supporting additional language models could be a valuable improvement. This could include incorporating other writing systems such as Katakana or even other languages with different scripts, like Chinese characters. By offering a wider variety of language models, the game could be used not only by learners of Japanese but also by those interested in other languages with complex writing systems, making the game more versatile and educational for different linguistic backgrounds.

6.6 Issues and Solutions

Poor OCR Recognition

The OCR system initially struggled with recognizing certain Hiragana characters, which impacted the accuracy of character recognition in the game. This was mainly due to the limitations of the default Tessdata and the lack of custom training data. To improve this, the threshold for OCR confidence was reduced, making the system more forgiving in its recognition. Additionally, a whitelist was implemented at each level to prioritize specific characters, ensuring the OCR system focused on the most relevant characters for the current stage. This helped improve the overall recognition rate, although challenges with accuracy remained for some characters.

Writing in VR Lacks Hand Support

Since the game requires players to write characters in VR, there was no proper hand support, making the writing experience awkward and challenging. This issue was addressed by adding haptic feedback to the VR controllers. The haptic feedback provided vibrations when the player drew characters, simulating the tactile sensation of writing and helping to improve the immersion and player experience. This solution, while effective, could be further enhanced in the future by incorporating more advanced hand tracking technology.

Limited Asset Quality and Immersion

The visual quality of the game was limited due to the absence of high-quality assets. The available assets did not meet the desired standards for immersion, affecting the overall aesthetic of the game. To address this, time was spent sourcing free assets from external platforms, such as OpenGameArt and other free asset libraries, instead of relying solely on the Unity Asset Store. While this solution helped, the game would greatly benefit from investing in higher-quality assets for future updates.

Technical Challenges with VR Setup and Equipment

A significant challenge during development was the lack of access to VR devices outside of the school lab. This limited testing and debugging time, as the devices had to be borrowed from the school and used in the lab, which was not ideal for continuous development. To resolve this, it was necessary to schedule dedicated time in the lab, although this resulted in delays. Future development would benefit from having more consistent access to VR devices, either through dedicated hardware or cloud-based VR testing solutions.

Lack of Customizable Language Models

Due to time constraints and limited resources, the project was unable to create custom language models for OCR recognition. The game had to rely on the default Tessdata provided by Tesseract, which did not offer optimal accuracy for the Japanese Hiragana characters. In future iterations, it would be beneficial to create and integrate custom language models to improve OCR accuracy, allowing the system to better recognize specific characters and handle more complex recognition tasks. This would enhance the educational aspect of the game, providing a more effective learning experience for users.

References

1) Roslina Mamat, Rokiah Paee, Roswati Abdul Rashid, & Normah Ahmad. (2022, May). VTubers and anime culture: A case study of Japanese learners in two public universities in Malaysia. ResearchGate. https://www.researchgate.net/publication/360997811_VTubers_and_anime_culture_A_case_study_of_Japanese_learners_in_two_public_universities_in_Malaysia
2) Solomon Sunday Oyelere1, Nacir Bouali, Rogers Kaliisa, George Obaido, Abdullahi Abubakar Yunusa, & Ebunayo R. Jimoh. (2020). Exploring the trends of educational virtual reality games: a systematic review of empirical studies. SpringerOpen. https://link.springer.com/content/pdf/10.1186/s40561-020-00142-7.pdf
3) Valem Tutorials. (2023, September 20). Let's make a VR Game Part 9 - Start menu [Video]. YouTube. https://www.youtube.com/watch?v=6WfowlZ51i8
4) Valem Tutorials. (2024, February 25). Learn VR Development in 3 Hours - Unity VR Tutorial Complete Course [Video]. YouTube. https://www.youtube.com/watch?v=YBQ_ps6e71k
5) Neelarghya. (n.d.). GitHub - Neelarghya/tesseract-unity: Standalone OCR plugin for Unity using Tesseract. GitHub. https://github.com/Neelarghya/tesseract-unity
6) Neelarghya. (2021b, December 13). Offline OCR using Tesseract in Unity. . . Part 1 - XRPractices - Medium. Medium. https://medium.com/xrpractices/offline-ocr-using-tesseract-in-unity-part-1-b9a717ac7bcb
7) Unity wrapper - Google Tesseract OCR. (2017, November 18). Unity Discussions. https://discussions.unity.com/t/unity-wrapper-google-tesseract-ocr/683155
8) Sketchfab. (n.d.). Japanese Classroom - Download Free 3D model by T I A N (@Tian96). https://sketchfab.com/3d-models/japanese-classroom-2a1e3b294c1e4e91bed794bfa520c4f4
9) The Architect - Asset Store. (n.d.). Unity Asset Store. https://assetstore.unity.com/publishers/40464

This page is intentionally left blank to indicate the back cover. Ensure that the back cover is black in color.