Thesis Dissertation

VOICE CONTROLLED GAME

Eirinaios Konstantinou

UNIVERSITY OF CYPRUS

COMPUTER SCIENCE DEPARTMENT

May 2026

UNIVERSITY OF CYPRUS

COMPUTER SCIENCE DEPARTMENT

VOICE CONTROLLED GAME

Eirinaios Konstantinou

Supervisor
Andreas Aristidou

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelor of Computer Science at the University of Cyprus

I declare that this dissertation and any accompanying software are my own original work, conducted in full compliance with the University of Cyprus Code of Conduct for Research (https://www.ucy.ac.cy/legislation/kodikas-deontologias-kai-politikes/), and with full accountability on my part for the accuracy, integrity, and validity of all content.

May 2026

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Andreas Aristidou, for his guidance, feedback, and continued support throughout the development of this thesis. His insights and direction were invaluable in shaping both the design and research aspects of this work.

I am also deeply thankful to the 20 participants who took the time to play through the game and share their honest feedback. This study would not have been possible without their willingness to engage with a new and sometimes unfamiliar form of interaction, and their input formed the foundation of the evaluation presented in this thesis.

Finally, I would like to thank my family and friends for their patience and unwavering support throughout my studies.

Abstract

Voice interaction represents one of the most natural forms of human communication, yet it remains underexplored in the gaming industry, where keyboards, mice, and controllers continue to dominate. While previous attempts at voice-controlled games have relied on predefined command grammars that constrain how players express their intent, recent advances in Large Language Models (LLMs) make it possible to interpret open-ended natural language. This thesis investigates whether voice controls, powered by modern LLMs, can serve as a viable alternative or complement to traditional control schemes in video games.

To explore this question, we designed and implemented a 2D platformer in which players issue commands using natural speech. The game features three levels, each tailored to highlight the strengths and weaknesses of a specific control scheme: a puzzle-oriented level testing voice or traditional controls, a fast-paced platforming level played twice with each scheme, and a boss battle that combines both into a hybrid control scheme. The voice pipeline consists of a local automatic speech recognition module based on OpenAI's Whisper and a cloud-based LLM module using Google's Gemini-2.5-flash, which interprets player intent and maps it to in-game actions through a structured reasoning framework.

A user study with 20 participants compared the three control schemes through gameplay metrics and a post-play survey. The results show that voice controls performed comparably to traditional controls in the puzzle scenario, with participants reporting similar enjoyment and slightly higher immersion despite some added frustration. In the fast-paced scenario, traditional controls clearly outperformed voice controls, with participants citing the latency between speaking a command and seeing it executed as the main obstacle. The hybrid scheme proved playable but did not fully combine the strengths of both methods, as voice commands tied to time-critical actions inherited the same latency issues. Overall, the findings suggest that voice controls are best suited for slower, thought-driven gameplay such as puzzles and exploration, where players have time to formulate spoken commands and engage with the world through natural language.

Content

Chapter 1

Introduction

1.1 Introduction to the topic

Video games are one of the most popular forms of entertainment today, offering immersive and interactive experiences to a wide range of audiences. Players interact with games through various input devices, with keyboards, mice and game controllers being the most common. In addition, other forms of input include touchscreens for mobile devices, virtual reality controllers for immersive environments, and specialized hardware such as steering wheels for racing games. Despite this wide range of interaction methods, one form of input that remains relatively underexplored in the gaming industry is voice interaction.

Voice interaction represents one of the most natural forms of human communication and has long been studied in both research and entertainment contexts. However, as discussed in [1], voice controls have historically struggled to achieve sustained acceptance in the gaming industry. The interest of voice interaction has repeatedly increased with each technological advancement, only to decline again when technical or usability limitations became apparent. At the same time, the dominance and reliability of traditional input devices such as keyboards, mice, and controllers have further limited the widespread adoption of voice-based control schemes.

Recent advances in speech recognition and natural language processing have significantly improved transcription accuracy, contextual understanding, and system responsiveness [2]. These developments make voice-based controls a more viable and increasingly relevant interaction scheme in modern interactive systems.

This thesis explores the potential of voice interaction as a control method in video games. Specifically, our work investigates how voice input compares with traditional control schemes across different gameplay scenarios and examines how natural language can be used as an input in interactive environments. Furthermore, the study analyzes how different control schemes influence player experience and immersion.

1.2 Motivation

The emergence of Large Language Models (LLMs) has significantly advanced natural language processing by enabling systems to interpret open-ended instructions and reason about context [3]. This development creates new opportunities for integrating LLMs into voice-control pipelines, allowing systems to move beyond predefined command grammars towards a more flexible and adaptive interaction.

At the same time, traditional control schemes continue to dominate digital games due to their responsiveness, precision and familiarity. This dominance raises important questions regarding the practical role of voice interaction in games and motivates an investigation into its strengths and limitations compared to conventional input methods.

Furthermore, several modern games have experimented with hybrid control schemes that combine voice input with traditional controls [4, 5]. However, these implementations typically rely on predefined command sets rather than open-ended natural language interpretation. This highlights an opportunity to explore whether recent advances in natural language processing can enable more natural and expressive forms of voice interaction in interactive environments.

1.3 Contribution

The primary objective of this thesis is to design, implement, and evaluate a voice-controlled game and to compare three interaction schemes: traditional controls, voice-controls, and a hybrid approach combining both.

Unlike traditional voice-control pipelines which depend on predefined command grammars and require users to follow specific rules, this project prioritizes natural user communication. Instead of restricting players to a fixed command structure, the system allows them to speak freely, using natural language, similar to how they would communicate in a human conversation.

This work aims to address the following research questions:

- RQ1: Do voice controls immerse the player more in the game rather than traditional controls?

- RQ2: Do voice controls perform better or worse than traditional controls in exploration and fast-paced gameplay scenarios?
- RQ3: Does a hybrid control scheme combine the advantages of traditional and voice controls?

By exploring these questions, this thesis contributes to the study of alternative interaction schemes in video games by examining the practical integration of modern Large Language Models into a real-time interactive system.

Chapter 2

Theoretical Background

2.1 The History of Voice Controls in Video Games

While the first speech recognition systems were developed in the 1950s [6], it was later in the 1970s that video games started being used as a research object. One of the earlies researched-based games developed with voice controls was *Voice-Chess* [7], which was built on the Hearsay-I speech recognition system. The game featured a turned-based structure with a well-defined command grammar of chess moves which made it ideal for the processing power of the time's technology. Later, voice interaction was explored as an alternative control scheme for users with disabilities as a domain of human-computer interaction. Research on that included voice actions that used volume or pitch instead of words for interacting [8].

As hardware capable of supporting voice interaction became more widely available, early implementations of voice controls began appearing in commercial video games. These systems were typically relied on predefined commands mapped to specific in-game actions. For example, *Nintendogs* [9] let the player command their virtual dog with phrases like 'sit' or 'roll over'. Similarly, the *Phoenix Wrigh a* [10] series allowed players to shout "Objection!" to interrupt the court.

Other titles incorporated voice input as a supplementary control mechanism. *Tomb Raider: Definitive Edition* [11] enabled the player to perform actions such as pausing the game, opening the map (by saying "show map"), or switching weapons by speaking their names. In *The Legends of Zelda: Phantom Hourglass* [12] players could blow into their microphones to extinguish torches and spin windmills. However, voice interaction in early games often included simple actions or alternative ways to perform actions that are already possible with the primary control scheme [1].

Voice interaction has also been used to enhance immersion. For instance, *Alien: Isolation* [13], a stealth horror game, introduced an optional 'noise detection" mode that used the player's microphone to detect any moderately loud noise, alerting the enemies to the

player's presence. Another example is the stealth game *Splinter Cell: Blacklist* [14] which allows the player to lure and distract enemies through voice commands. As discussed in [15], these mechanics strengthen the connection between the player and the game, contributing to an increased sense of presence and immersion.

2.2 Voice Interaction Systems

In typical voice-controlled systems, interaction follows a multi-stage processing pipeline. First, an audio capture device records the user's spoken input. Then, the audio is processed by a speech recognition system which converts it into textual form. Subsequently, a Natural Language Understanding (NLU) system interprets the transcribed text and maps it to predefined system actions, which are then executed by the application.

Traditional implementations of this pipeline rely heavily on fixed command grammars and rule-based mappings. While this approach ensures predictable behavior, it limits flexibility and requires users to adapt to rigid command structures. As a result, users often adapt the way they speak by simplifying their language, using specific keywords, or mentally preparing the instructions before issuing them. This behavior suggests that many users perceive voice-controlled systems as limited in their ability to understand natural language [16].

Furthermore, as discussed in [17], voice interaction can affect users' sense of agency. Due to the spatial and temporal separation between speaking a command and observing its execution, users may perceive voice interfaces as less immediate and less controllable compared to traditional input devices. Additionally, research indicates that users tend to overestimate the time required for voice systems to process and execute their instructions, which contributes to the perception of reduced responsiveness [17].

The technical aspect of voice controlled systems is also another common reason for dissatisfaction. Variation in language, accent, and dialect, can affect the transcription accuracy. In combination with the many possible ways of phrasing the same instruction, these factors can lead to incorrect interpretations or unintended actions. Without effective error handing, such failures can negatively impact usability and overall experience [18].

Chapter 3

Methodology

3.1 Game Design

The game includes 3 levels with different gameplay scenarios, each highlighting the strengths and weaknesses of each control scheme. We implemented the levels in a 2D perspective with the player giving instructions on the character with natural language. Overall, in the game the player was not given the list of available actions the character could do and was tasked with discovering them by examining the environment and prompting the character to try different things. In addition, some levels are played with both traditional and voice controls in order to get data to compare them.

3.1.1 Level 1

The first scenario is a puzzle-oriented level where the player needs to acquire a password to get inside the house. To complete this level, the player must activate three runes hidden across the 3 different areas of the level, which together reveal the password.

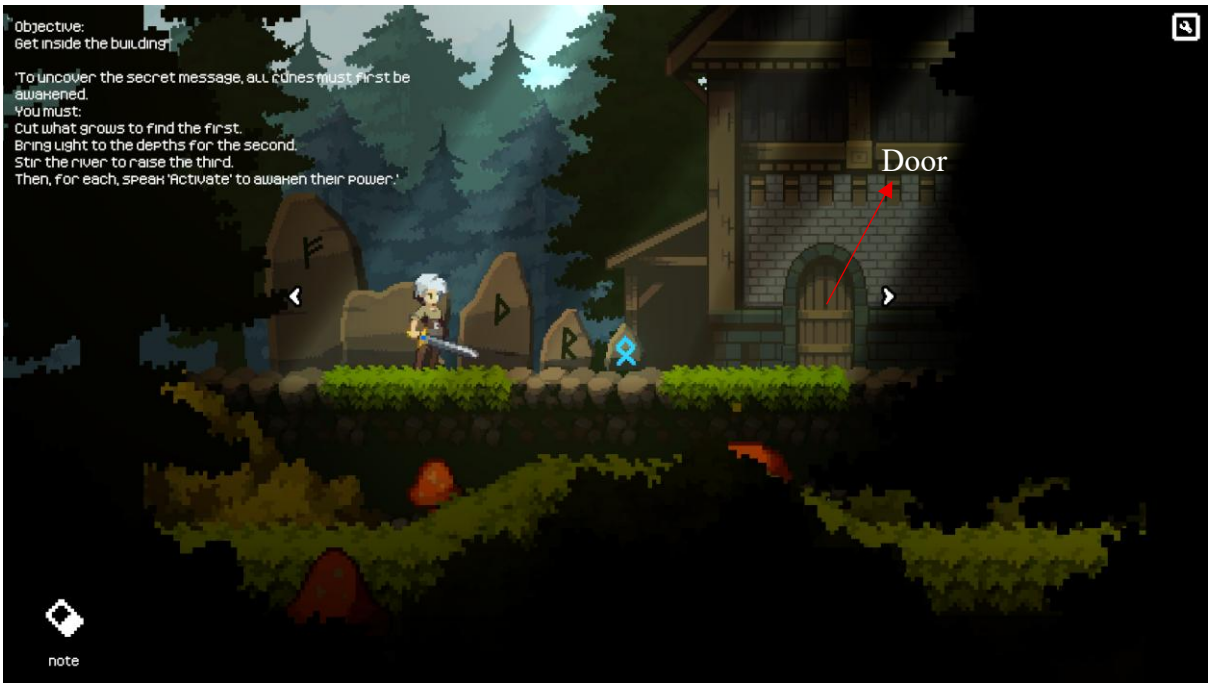

Figure 3.1 – Area 2. The starting area of the level. The objective of the level is to get inside the house through the door. To do that they need to reveal the password by activating all three runes shown in that area.

If the player attempts to interact with the door before solving the puzzle, a message will appear asking them for the password (Figure 3.2).

Figure 3.2 – The message that appears when the player tries to interact with the door or get inside the house without solving the puzzle.

Starting Step

At the starting point, a note containing instructions guides the player through the puzzle. Initial playtesting revealed that players struggled to notice the note unprompted, making it necessary to add an on-screen hint directing them to look around (Figure 3.3).

Once the player issues a "look around" command, they discover the note on the ground, pick it up, and read it. Its contents then appear persistently on the left side of the screen for reference throughout the level.

Figure 3.3 – At the start of the level, a hint is displayed to guide the player in initiating the puzzle. The hint prompts the player to give a “look around” command.

Rune 1

The clue *"Cut what grows to find the first"* directs the player to Area 1, where the first rune is hidden behind a bush (Figure 3.4). The player issues a "cut bush" command to reveal it, then speaks "activate" to complete the activation, which changes the rune's color to blue as visual confirmation (Figure 3.5).

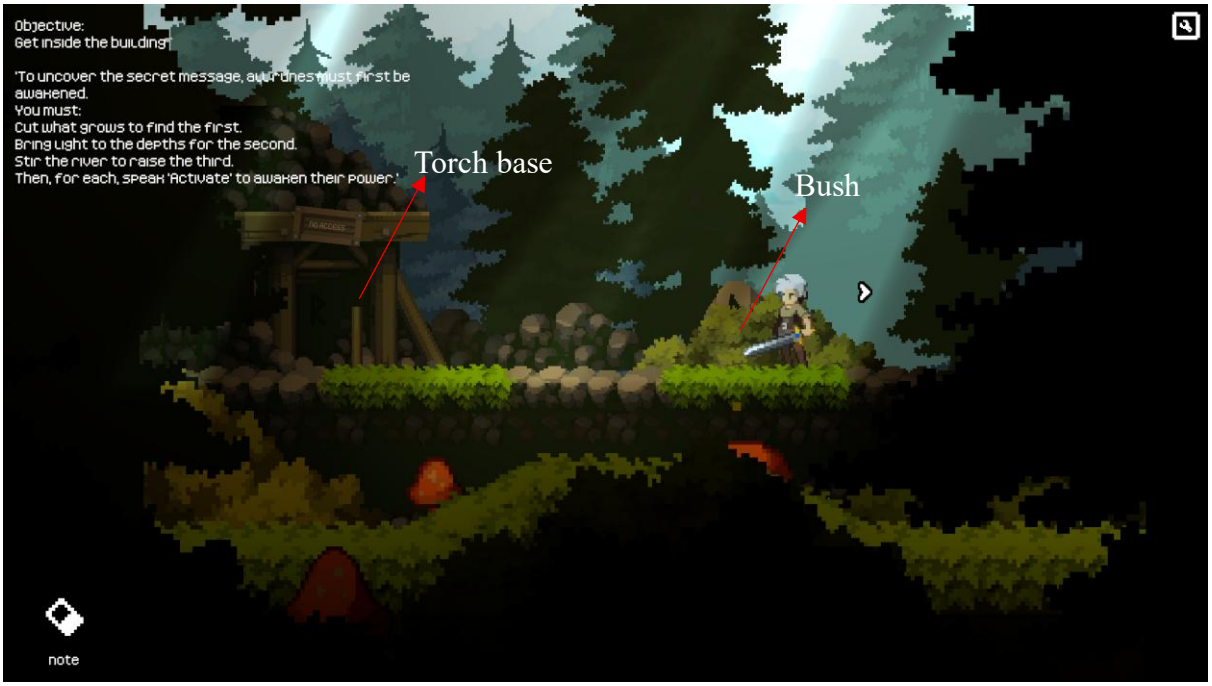

Figure 3.4 – Area 1. This area contains two hidden runes that the player must discover: one located behind a bush and another inside a cave. To reveal the first rune, the player must give a "cut bush" command, while the second requires placing a torch obtained from Area 3.

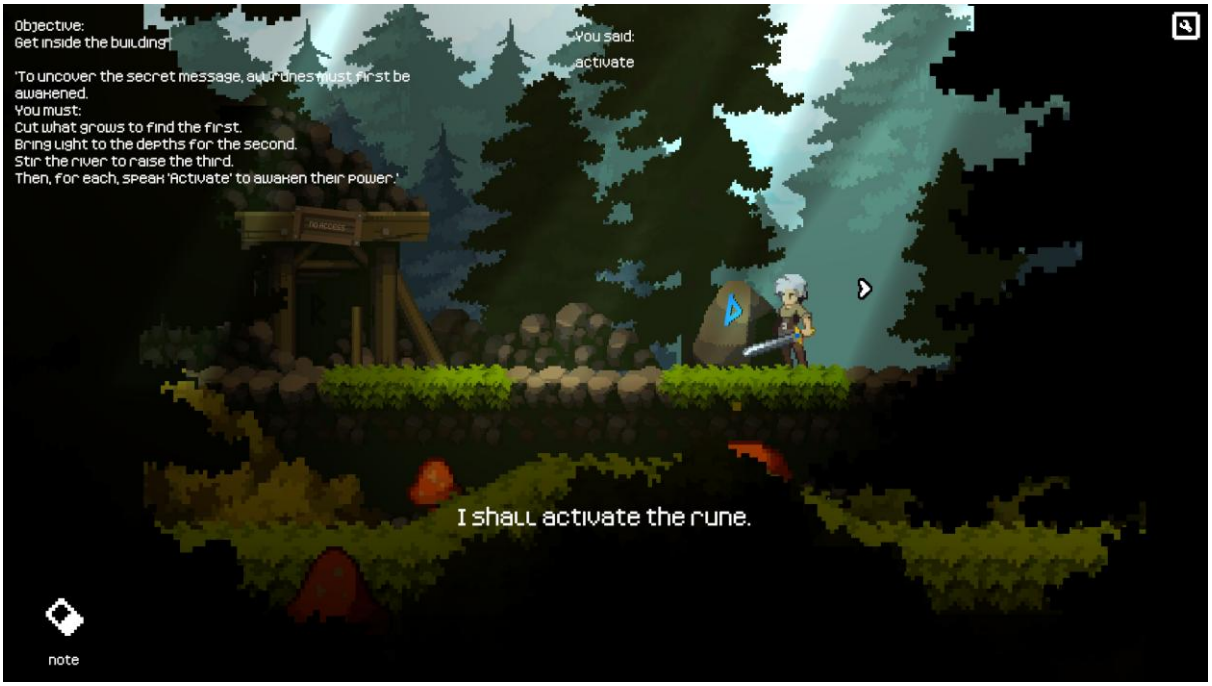

Figure 3.5 – To activate a revealed rune, the player must give the command "activate," which changes the symbol's color to blue to indicate successful activation.

Rune 2

The clue *"Bring light to the depths for the second"* requires the player to retrieve a torch from Area 3 and place it on a torch base in Area 1. To reach Area 3, the player must cross a river in Area 2 by jumping over it. Placing the torch illuminates the cave and reveals the second rune, which the player activates using the same "activate" command as before (Figure 3.6).

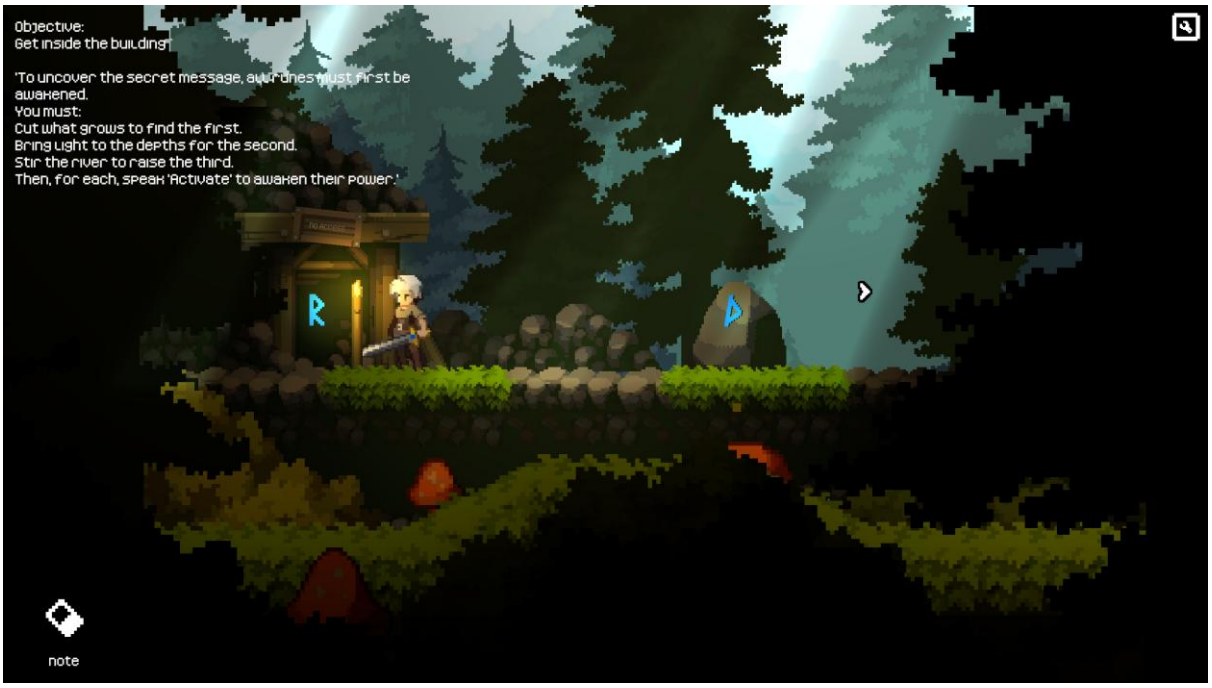

Figure 3.6 – To activate the second rune the player must pick up the torch from area 3 and place it on the torch base. After that they can say 'activate'.

Rune 3

The clue *"Stir the river to raise the third"* points the player to the river in Area 3. A sign beside a pile of rocks draws attention to the interaction point (Figure 3.7).

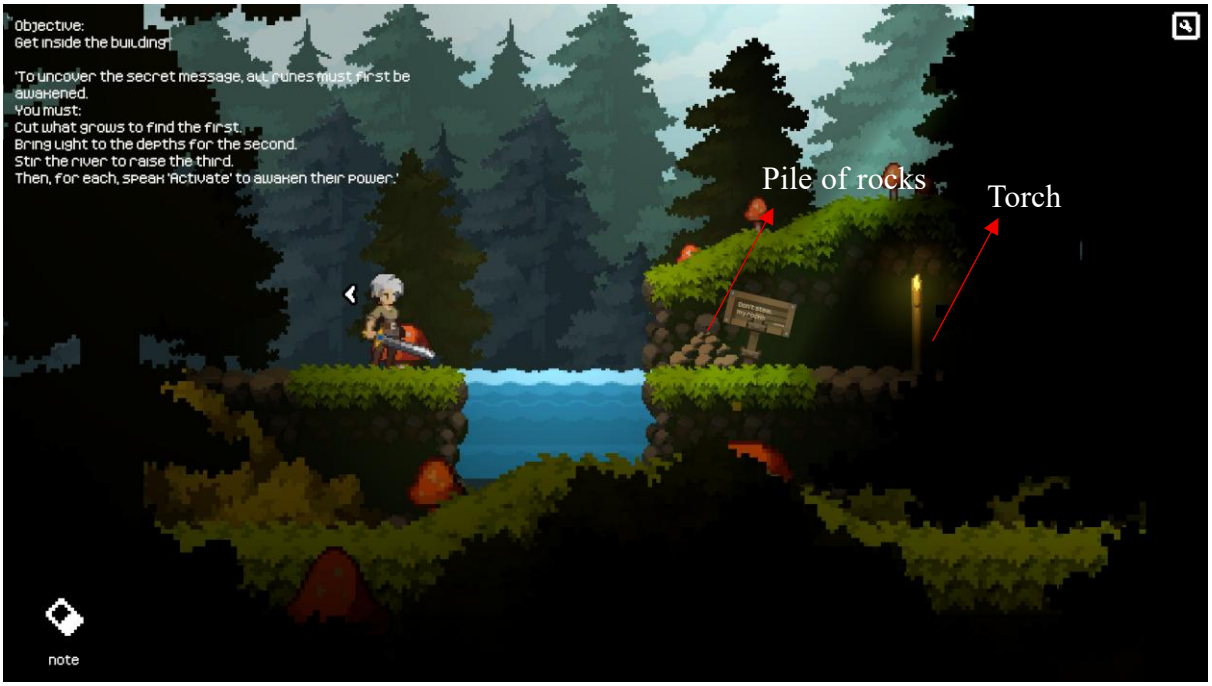

Figure 3.7 – Area 3. Here there is a river in the middle that the player must jump over. On the other side there is a sign that is hinting to grab and throw the rock in the river, as well as the torch needed for activating rune 2.

The player picks up a rock and throws it into the river, causing the third rune to rise from the water. As with the previous runes, the player then activates it using the "activate" command (Figure 3.8).

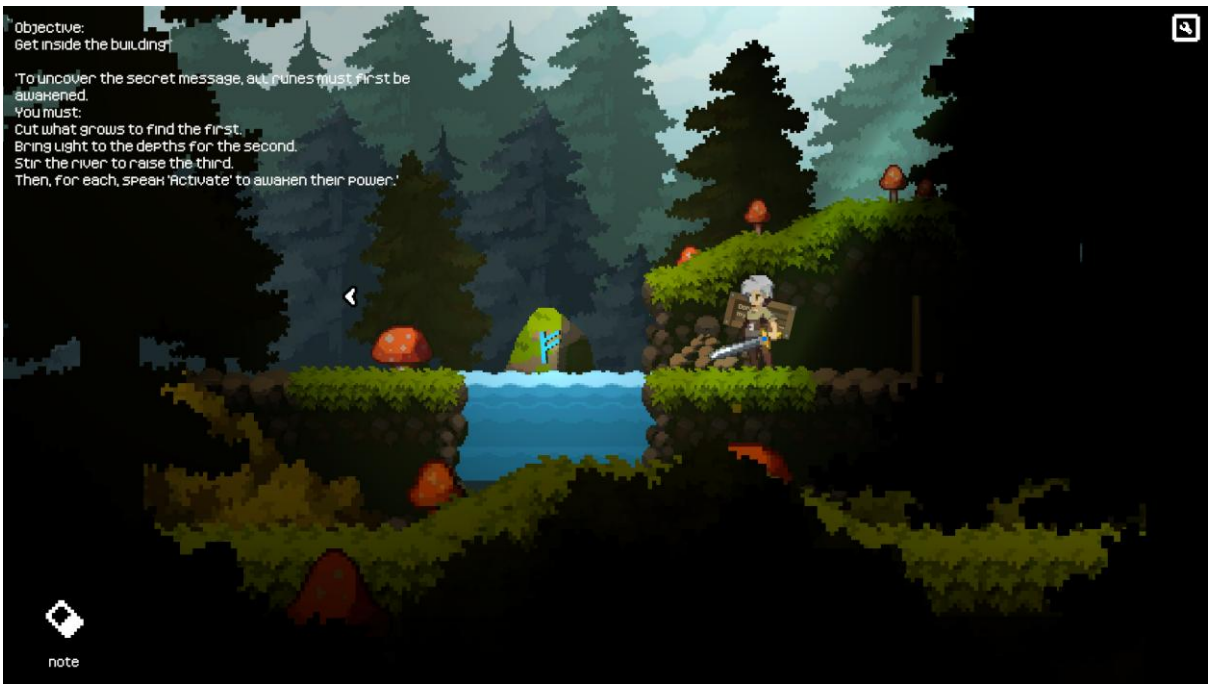

Figure 3.8 – The player needs to pick up a rock and throw it in the river. Then the rune will rise from the river and they will be able to activate it.

Revealing The Password

Once all three runes have been activated, a message will appear on a rock in area 2. The player needs to tell the character to read it in order to learn the password. After that, the player can navigate to the door and speak the password in order to get inside and complete the level (Figure 3.9).

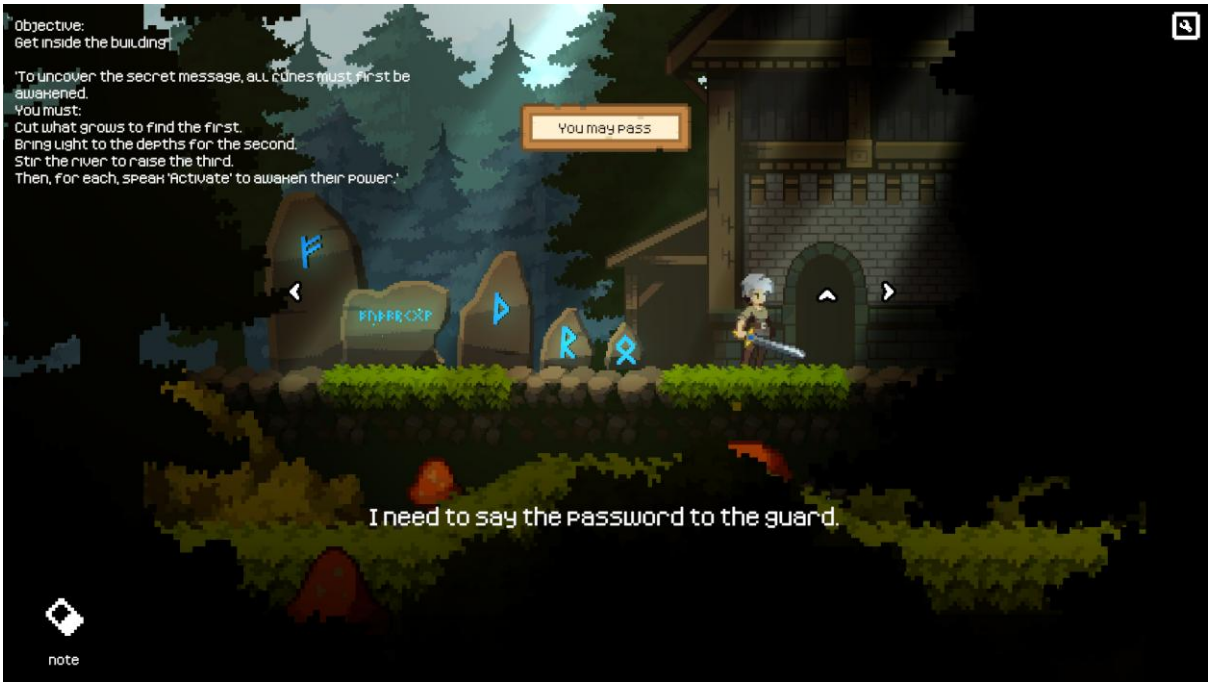

Figure 3.9 – After all runes have been activated, a rock in area 2 will reveal the password. The player needs to tell the character to read it and once they know it they can tell navigate to the door and speak it to compete the level.

3.1.2 Level 2

The scenario of this level focuses on time-based interactions and obstacles that must be passed in limited time. It contains 3 areas that challenge the player's timing and are focused on the strengths of traditional controls and weaknesses of voice controls.

Area 1

In this area the player needs to swing across the chains to reach the other side. They can do that by giving a command like 'jump and grab the chain'. In case they fail, they are respawned to the start.

Figure 3.10 - Area 1. There are two chains swinging back. The player needs to swing across by giving commands like "jump and grab the chain".

Area 2

For this area, the player needs to time their movement in order to avoid hitting the statues.

Figure 3.11 - Area 2. There are three status moving up and down at different intervals. The player must time their "move right" command in order to pass by all three without getting hit.

Area 3

The last area includes a combination of the previous obstacles and introduces a time-limited challenge. The player needs to open the gate by saying ‘activate the gate mechanism’ and then pass the obstacles before it closes. This area challenges the way players give their commands by requiring them to say multiple instructions at once. For example, while they are hanging from the chain, to pass the statue they need to say something like ‘jump and move right’.

Figure 3.12 - Area 3. This area combines the obstacles of the previous two areas. The player needs to open the gate, swing with the chain, pass the statue, and go inside before it closes.

3.1.3 Level 3

The final level of the game is played using hybrid controls (both keyboard and voice) and features a boss battle (Figure 3.13). This scenario tests the functionality and usability of the hybrid control scheme by having the player control the movement of the character with the keyboard (WASD) and the interactions with the environment with their voice.

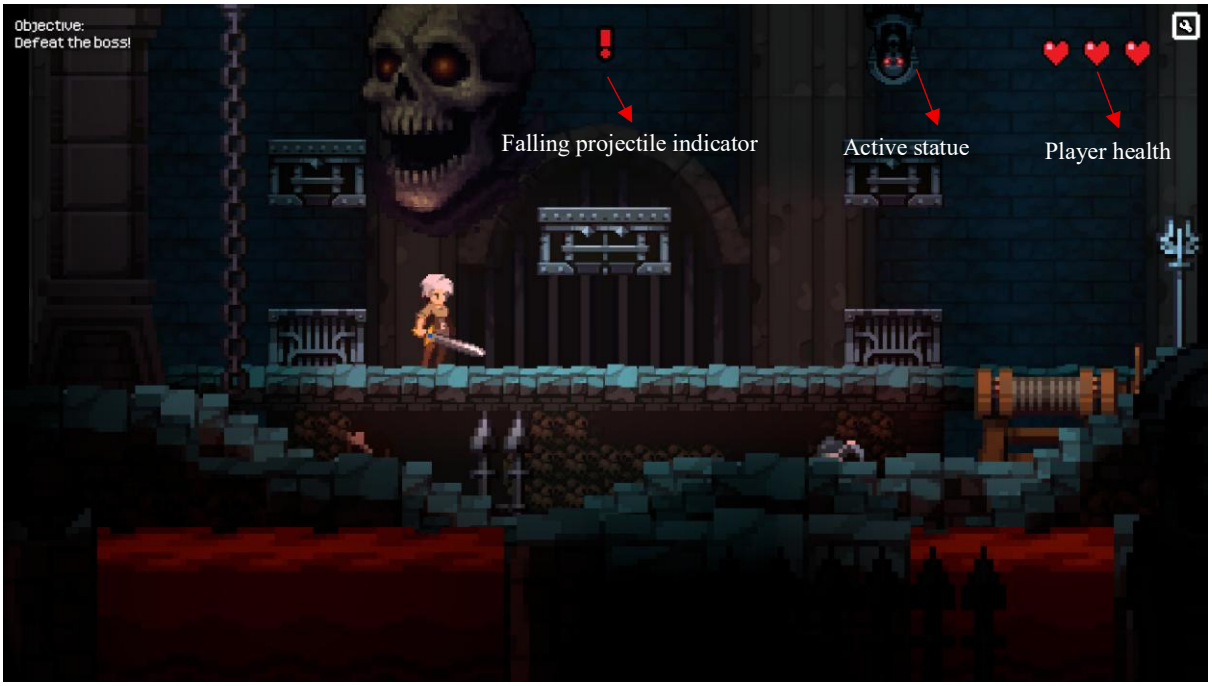

Figure 3.13 - The boss battle area. The falling projectile indicator (red exclamation mark) warns the player of an incoming attack, giving them time to respond. The active statue marks the current interaction point for the Attack action. Player health is displayed in the top-right corner as three hearts.

Voice Actions

The player had 2 actions available which were explicitly shown at the start. The first is *Attack* which can be used in front of active statues to deal damage to the boss and the second is *Block*, used to protect the character from the falling rocks.

Boss Design

The boss has 4 health points, meaning that if the player successfully activates the *Attack* action 4 times, they will win. When the boss loses their first 2 health points, the next phase activates and lava appears on the floor, making navigation harder, as touching it will instantly kill the character.

Falling Rocks

While the player is trying to defeat the boss, rocks fall periodically from the sky, dealing damage to them. The character has 3 health points, meaning when they get hit by rocks 3 times, they lose and need to restart the level. Because of the latency of speech to in-game commands, it was necessary to show indicators of incoming falling projectiles to warn the player and allow them time to use the *Block* action.

There are 2 different falling projectile attacks. Both attacks aim to be resolved by using voice or traditional controls. The symbol in Figure 3.14 (right) indicates that a large attack is coming which is only avoidable by using the Block action. The symbol in Figure 3.14 (left) indicates a small attack which is avoidable by just moving away with keyboard controls.

Figure 3.14 - Falling projectile indicators. The small attack indicator (left) warns the player of an incoming projectiles that can be avoided by moving away with the keyboard. The large attack indicator (right) warns of projectiles that requires the player to use the "Block" voice command.

3.2 Speech-to-Commands

The core mechanic of the game is a push-to-talk system that allows players to give instructions in natural language while holding a designated input key. We achieved this by designing a two-stage pipeline that translates speech into in-game actions (Figure 3.15). The pipeline consists of a speech-to-text (STT) module and a text-to-commands module based on an a Large Language Model LLM.

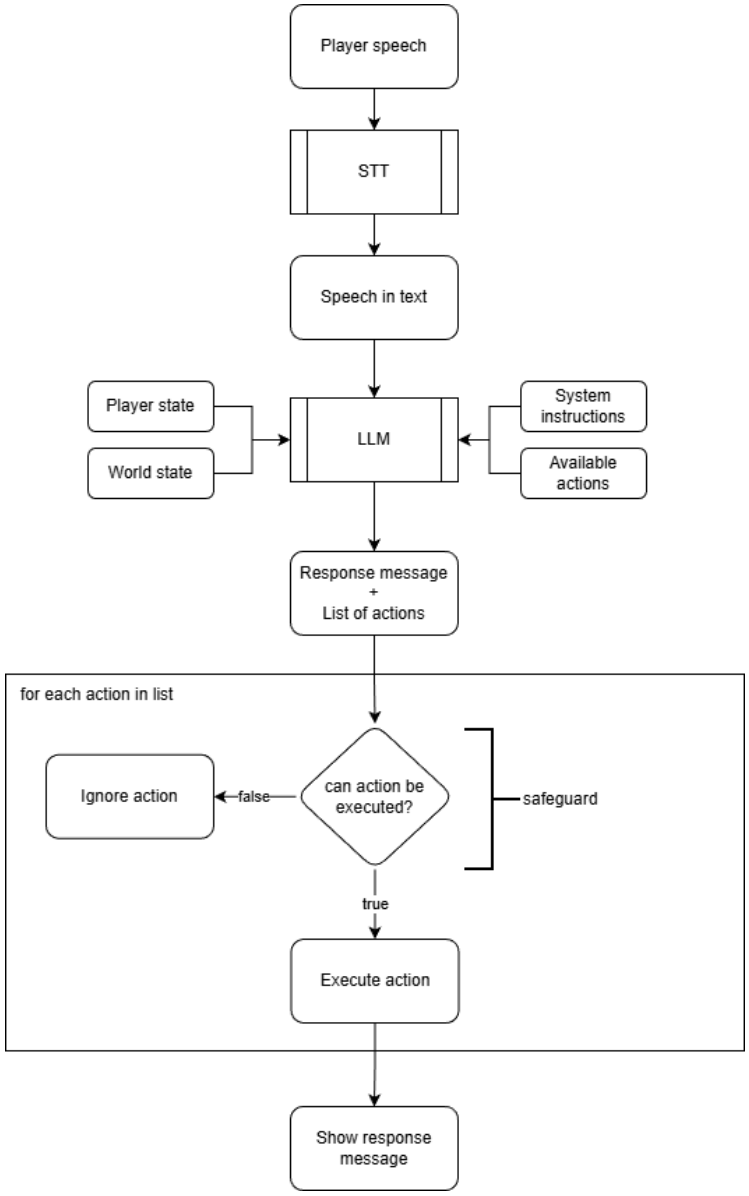

Figure 3.15 - Speech-to-in-game-commands Pipeline

3.2.1 Speech-to-Text Module

This Speech-to-Text module transcribes the player's spoken input into textual form. The resulting text serves as an input to the Text-to-Commands module and represents the first stage of the Speech-to-Commands pipeline.

3.2.2 Text-to-Commands Module

The text-to-commands module is responsible for translating the transcribed speech into in-game commands. This task requires semantic understanding and structured reasoning, as

the system must identify the player's intent, goals and determine a sequence of valid actions that achieves the desired outcome.

We based the overall architecture on the design patterns proposed in [19] , adapting them to support navigation, interaction and action execution. Given recent advancements in Large Language Models (LLMs), which demonstrate remarkable proficiency in natural language interpretation and structured reasoning [2], we selected an LLM to act as the intermediary between player speech and in-game actions.

To generate a list of actions that satisfy the player goals, we implemented the Reasoning via Planning (RAP) framework [20], which enables the LLM to act both as a world model and a reasoning agent. As shown in Figure 3.16, each prompt provided to the model contained:

- The player's transcribed instruction
- The current world state

The world state included the player's position and inventory, as well as the information about each tile in the area. To represent environmental information, we employed the Waypoints design pattern described on [19], assigning key identifiers to tiles. This structure allowed the model to interpret spatial references (e.g. "the flower next to me") within the current environment representation. Additionally, we provided explicit system-level instructions to guide the reasoning process of the model as well as the list of available actions for the current level.

```
{
    "query" : "Go to that door",
    "playerState":{
        "tileIndex" : 0,
        "inventory" : ["note"]
    },
    "areaState" : {
        "tiles" : [
            {
                "tileIndex" : 0,
                "references" : ["rock","flower"],
            },
            {
                "tileIndex" : 1,
                "references" : ["house","door"],
            },
            ...
        ]
    }
}
```

Figure 3.16 – LLM input example, including the player's query, player state and world state

Safeguards

Natural language instructions can often be ambiguous, underspecified, or unrealistic, creating challenges for systems that try to interpret them [21]. When an LLM processes such instructions, it is possible to interpret them incorrectly, leading to actions that are invalid in the current context. For example a player may attempt to pick up an item that does not exist at their current location. Even when provided with the world state, the model may still generate a "pick-up" action due to ambiguity or imperfect reasoning. To prevent invalid behavior, we implemented execution-level safeguards. Rather than relying solely on prompt constrains, we enforced validation directly within each action. Before execution, each action verifies its preconditions against the current game state. If the required conditions are not satisfied (e.g. the referenced item is absent), the action will be ignored.

Chapter 4

Implementation

4.1 Overview

The system we developed enables the player to control the game using natural language. Player speech is captured in real time, transcribed into text, interpreted using a large language model (LLM) and mapped to in-game commands. The implementation consists of a modular pipeline architecture which allows for alternative implementations for both speech recognition and command interpretation and enables easy experimentation.

4.2 Unity Engine

For the development of the game we used the Unity game engine, version 2022.3.25f1 with C# as the programming language. Unity provides a component-based architecture that enables modular game logic and seamless integration with external services.

The core components used were game objects for game entities such as the player, environment, interactable objects and user interface. In addition, the game's world was constructed using Unity's Tilemap system for a grid-based environment representation. Pixel-art assets were obtained from itch.io , while sound effects from Pixabay.

The functionality of the game was handled through custom C# scripts and the audio input was captured using the Microphone service of C#.

4.3 Local Backend Integration

To support local speech recognition and LLM processing, communication between Unity and locally running backend services is handled using the NativeWebSocket library. This library provides WebSocket-based bidirectional communication with low latency data exchange between the game client and local python servers.

4.4 Cloud Backend Integration

Firebase was used as a backend-as-a-service to support communication with cloud-based LLMs. To integrate it with Unity we used the Firebase Software Development Kit (SDK) which is a bundle of libraries and API wrappers that let the game easily communicate with services like Google Cloud.

More specifically, we used the Firebase.AI library which allowed the use of Google's Gemini models. The GetGenerativeModel method was used to establish the model connection and GenerateContentAsync(prompt) was used to receive model responses.

4.5 Speech-to-text Module

This speech-to-text module is responsible for transcribing player speech into text. The primary design goal of this module was to balance transcription accuracy with low latency in order to preserve a responsive gameplay experience.

4.5.1 Cloud-Based STT

In an initial implementation, we used Google's Gemini-2.5-flash LLM via Firebase to perform the speech transcription. Audio was transmitted to the model in Waveform (WAV) format and processed remotely, with the transcribed text returned to the game (Code 4.1). This method was straightforward to set up with minimal local setup but produced a noticeable latency due to network communication and cloud processing.

```
var model = ai.GetGenerativeModel("gemini-2.5-flash");
var prompt = ModelContent.Text("Transcribe what's said in this audio
recording.");
var audio = ModelContent.InlineData("audio/wav", audioData);

var response = await model.GenerateContentAsync(new[] { prompt, audio });
```

Code 4.1 - LLM Speech Recognition

4.5.2 Local ASR Server

As an alternative, we implemented a local automatic speech recognition (ASR) solution using OpenAI's Whisper model via the faster_whisper Python package. Faster_whisper offers increased speed and reduced memory usage compared to the original Whisper implementation. Whisper provides multiple model sizes with varying computational and memory requirements. For the implementation of the local ASR server, we selected the small model which offered a balance between transcription quality and speed while remaining lightweight for the user's hardware.

A local Python server is launched when the game starts and listens for control and data messages on dedicated ports. As illustrated in Figure 4.1, when the player initiates push-to-talk, audio data is streamed to the server, buffered, and transcribed once recording stops. Then, the result is returned to Unity. The torch library has also been used to detect GPU availability and accelerate performance when supported.

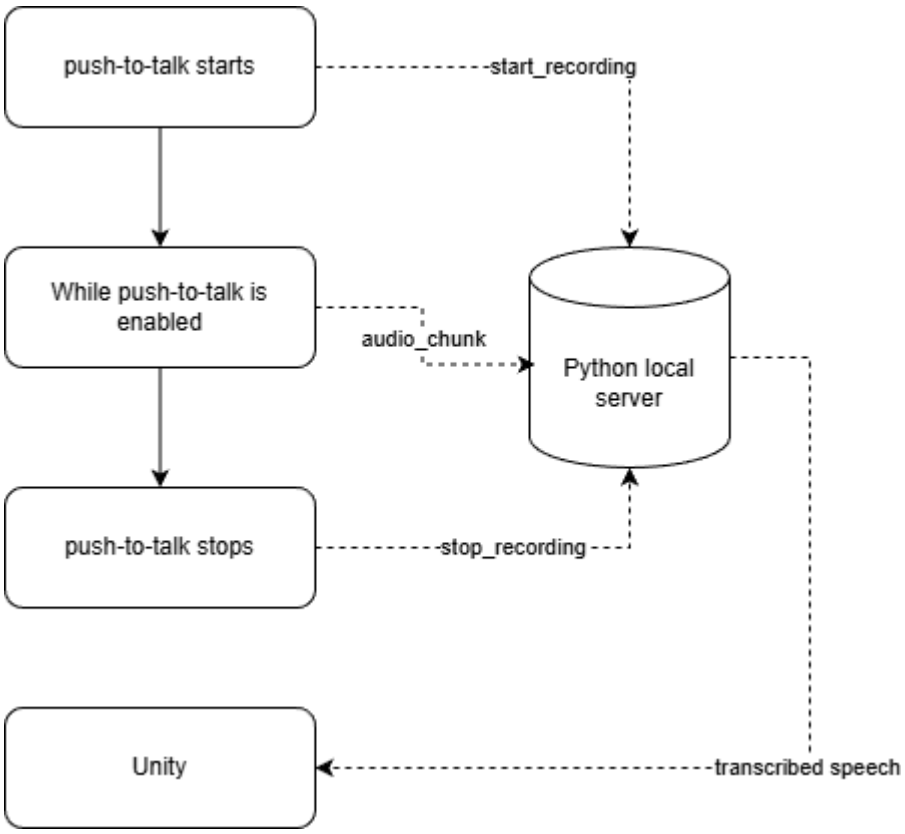

Figure 4.1 - STT Overview

4.5.3 Selected STT Approach

We chose to implement the local ASR system to the game. Despite it being dependent on the user's machine, this approach offered a good balance between transcription accuracy, responsiveness, and cost efficiency, as it runs entirely offline without relying on cloud services.

4.6 Text-to-Commands Module

The primary objective of this module was to reliably convert transcribed speech into in-game commands while maintaining low latency and enforcing a structured output format. To ensure deterministic parsing, we constrained the model to return responses in JavaScript Object Notation (JSON). Additionally, we imposed a responsive time limit to prevent excessive delays and preserve a response gameplay experience.

4.6.1 Cloud-Based LLM

We initially developed a cloud-based LLM implementation using the gemini-2.5-flash model via Firebase. To reduce operating costs, we optimized system instructions to

removed unnecessary information and disabled conversation history tracking. Each request is processed independently, ensuring predictable token usage and response times.

The model's configuration included the response type to be in JSON format, maximum output length of 2048 tokens for cost optimization, and a temperature value of 0.2 to limit the model's randomness (Code 4.2).

```
model = firebaseAI.GetGenerativeModel(
    modelName: "gemini-2.5-flash",
    systemInstruction: ModelContent.Text(instructions),
    generationConfig: new GenerationConfig(
        responseMimeType: "application/json",
        maxOutputTokens: 2048,
        temperature: 0.2f,
        candidateCount: 1
    )
);
```

Code 4.2 – Model Initialization

4.6.2 Local LLM

An alternative approach was implementing a Python server similar to the STT architecture. Models evaluated included gpt-oss-20b and qwen3-4b-thinking-2507, accessed via LM Studio and the lmstudio Python library. While local models offered a cost-free solution with offline execution, their performance and reasoning capabilities were strongly dependent on the user's machine and prompt length was limited to maintain responsiveness.

4.6.3 Selected LLM Approach

We chose to implement the cloud-based solution to the game. This approach, provided more reliable reasoning capabilities and consistent behavior across different machines, making it better for translating player intent in the game.

Chapter 5

Results & Evaluation

5.1 Research Process

To evaluate and compare voice and hybrid controls against traditional controls, we distributed the game to 20 people. Although system performance may vary depending on hardware specifications, only the speech recognition component operated locally on the user's machine. For this reason, we chose to distribute the game rather than conduct the study in a fully controlled environment.

Due to the importance of audio quality for speech recognition accuracy, participants were required to use a dedicated external microphone. Testing with built-in microphones (e.g. integrated laptop microphones) produced inconsistent results and reduced transcription reliability.

Because participants completed the study remotely and without supervision, it was necessary to incorporate technical support features within the game. The game included configurable settings such as microphone selection, audio controls and options to restart the level or reinitialize the speech recognition system in case of malfunction. Participants were also provided with our contact information in case they encountered technical issues that could not be resolved through the in-game settings.

The game consisted of three gameplay scenarios. The **first level** was designed as a puzzle-based task. To avoid solution bias, each participant completed this level only once, using either voice or traditional controls, determined randomly at the start of the game. The **second level** was a fast-paced experience that was completed twice: once using voice controls and once using traditional controls. Finally, the **third level** required participants to use the hybrid control scheme, combining voice and traditional input.

5.2 Participant Demographics

A total of 20 participants took part in the user study. As shown in Figure 5.1, the majority fell within the 25-34 age range (66.7%), followed by the 18-24 group (26.7%), with one participant aged under 18.

All participants were frequent gamers. As shown in Figure 5.2, 46.7% reported playing video games 4–6 times per week, 33.3% played daily, and 13.3% played 1–3 times per week, with the remaining 6.7% playing less than once a week. No participant reported never playing games. In terms of preferred genres (Figure 5.3), Shooter games were the most popular (66.7%), followed by Action and RPG titles, each selected by 60% of participants, suggesting a sample with a strong preference for fast-paced, interaction-heavy gameplay.

Regarding prior experience with the control schemes used in the study, participants were highly familiar with keyboard controls, with 73.3% rating their familiarity at 5 out of 5 (Figure 5.4). This is a relevant baseline given that Level 1 and Level 2 include a traditional keyboard control condition. In contrast, 66.7% of participants reported having never used voice controls in games or applications before, while 20% were unsure and only 13.3% had prior experience (Figure 5.5). This indicates that for most participants, the voice and hybrid control conditions represented a new interaction method.

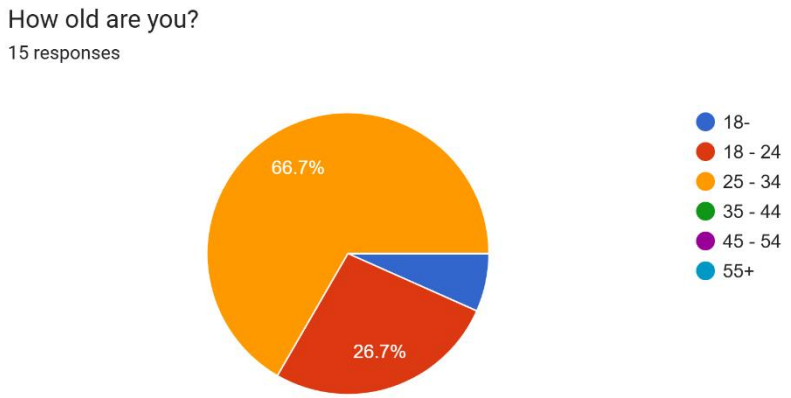

Figure 5.1 - Survey age data

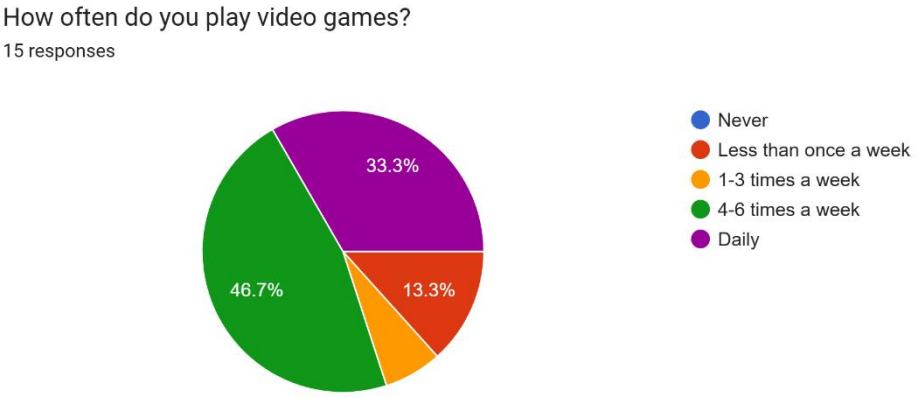

Figure 5.2 - Survey video game habits data

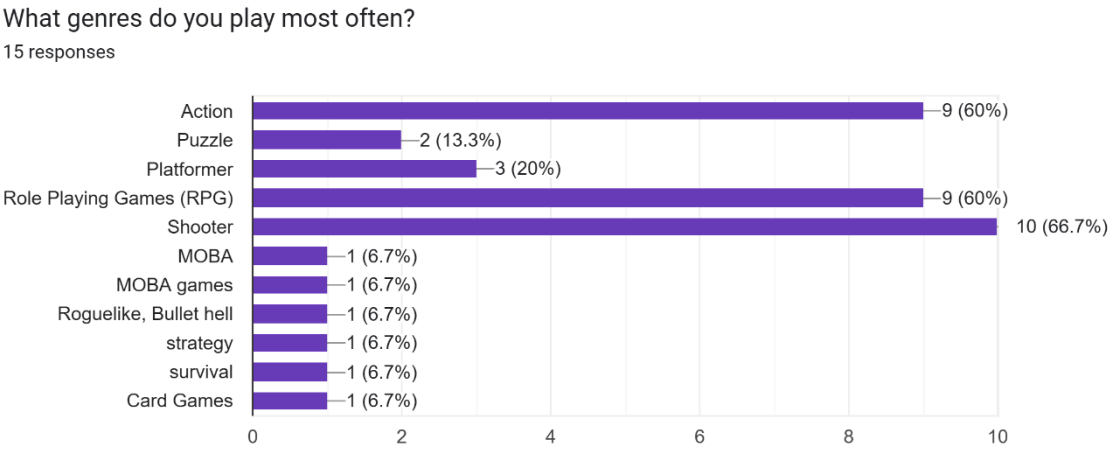

Figure 5.3 - Survey video game genre habits data

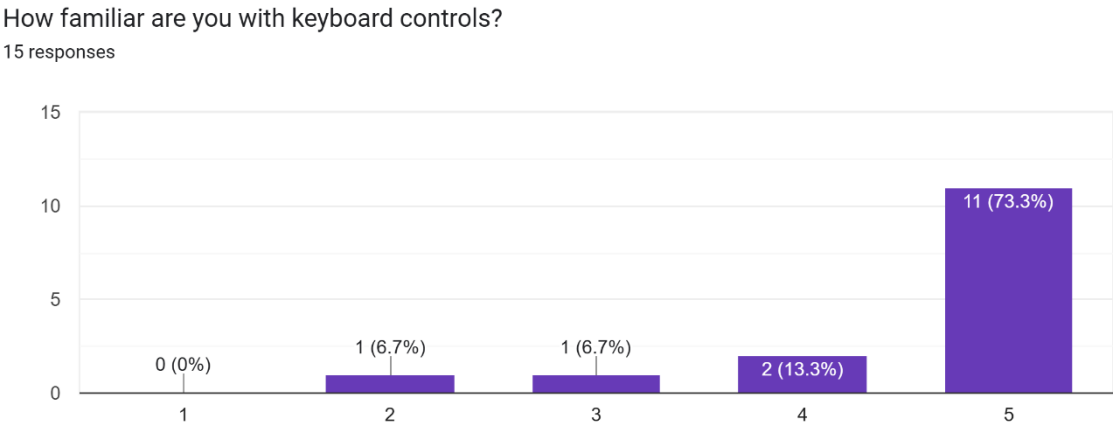

Figure 5.4 - Survey keyboard controls familiarity data

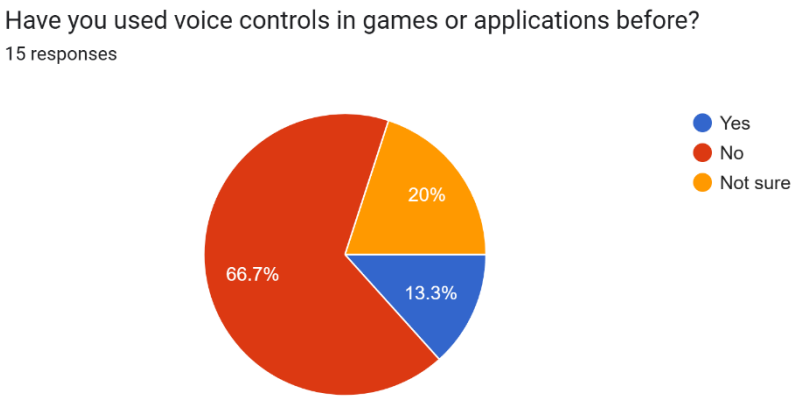

Figure 5.5 - Survey voice controls familiarity data

5.3 Data Collected

After completing all gameplay scenarios, participants filled out a questionnaire designed to evaluate their experience with each control scheme. The survey included closed-ended questions to measure factors such as frustration, enjoyment, immersion, and perceived usability. In addition, the game recorded performance metrics, including level completion time and the number of failed attempts in each level. Participants were instructed to report these values in the questionnaire. (Questionnaire link)

Table 5.1 presents the survey questions and their relation to the corresponding research questions. Additionally we used questions from the study in [22] which explores player immersion through a series of experimental methods and evaluation techniques.

No.	Question	Related Research Question
0	[Level 1] How much time did it take you to complete the level?	RQ2
1	[Level 1] The controls felt intuitive	RQ2
2	[Level 1] I could easily understand how to interact with the environment	RQ2
3	[Level 1] I felt in control of my character's actions	RQ2

4	[Level 1] I experienced frustration when trying to move the character	RQ1
5	[Level 1] I experienced frustration when trying to interact with the environment	RQ1
6	[Level 1] I had fun playing this level	RQ1
7	[Level 1] How immersed did you feel?	RQ1
8	[Level 2] Which control method felt more effective for this level?	RQ2
9	[Level 2] How much time did it take you to complete the level?	RQ2
10	[Level 2] How many failed attempts did you make?	RQ2
11	[Level 2] I experienced frustration when trying to react to the environment	RQ1
12	[Level 2] I was unaware of what was happening around me in the real world	RQ1
13	[Level 2] I had fun playing this level with voice controls	RQ1
14	[Level 2] How immersed did you feel?	RQ1
15	[Level 3] How much time did it take you to complete the level?	RQ3
16	[Level 3] How many failed attempts did you make?	RQ3
17	[Level 3] The controls felt intuitive	RQ3
18	[Level 3] The voice commands enhanced the experience	RQ3
19	[Level 3] I felt detached from the outside world.	RQ1
20	[Level 3] I had fun playing this level	RQ1 & RQ3
21	[Level 3] How immersed did you feel?	RQ1 & RQ3
22	Which control scheme made you feel more immersed	RQ1

23	Voice controls in games are better suited for: (multiple choice)	RQ2

Table 5.1 – Survey Questions with the related Research Questions

5.4 Survey Results

This section presents the results of the post-play survey completed by all 20 participants. Responses were collected on 5-point Likert scales (1 = strongly disagree, 5 = strongly agree) for most questions, while immersion was measured on a separate 1–10 scale.

5.4.1 Statistical Methods

Due to the small sample size (n = 20) and ordinal nature of Likert-scale data, this study relies on descriptive statistics and chi-square tests throughout the analysis. For categorical preference questions, such as which control scheme participants found most effective, chi-square tests determine whether the observed distribution of choices differs significantly from what random chance would predict, producing a p-value that represents the probability the observed pattern occurred by chance alone. We consider a result statistically significant when its p-value falls below the chosen threshold of $\alpha = 0.05$. For Likert-scale responses, we report the mean (M) and median (Mdn) to describe and compare group responses, where the median often better represents small samples with skewed distributions.

5.4.2 Level 1

In Level 1, participants were randomly assigned to voice controls (n = 11) or traditional controls (n = 9). Traditional control users rated the controls as more intuitive (M = 4.22 vs. M = 3.55) and reported a clearer understanding of how to interact with the environment (M = 4.67 vs. M = 3.27), as well as a stronger sense of control over their character's actions (M = 4.67 vs. M = 3.45). These differences suggest that familiarity with conventional input methods gave traditional control users a more confident experience from the outset.

The clearest differences appeared in frustration. Voice control users reported considerably higher frustration both when moving the character (M = 3.36 vs. M = 1.22) and when interacting with the environment (M = 3.18 vs. M = 1.33). This suggests that voice controls did not fundamentally prevent engagement with the puzzle itself, but the act of translating intent into spoken commands introduced friction that traditional input did not.

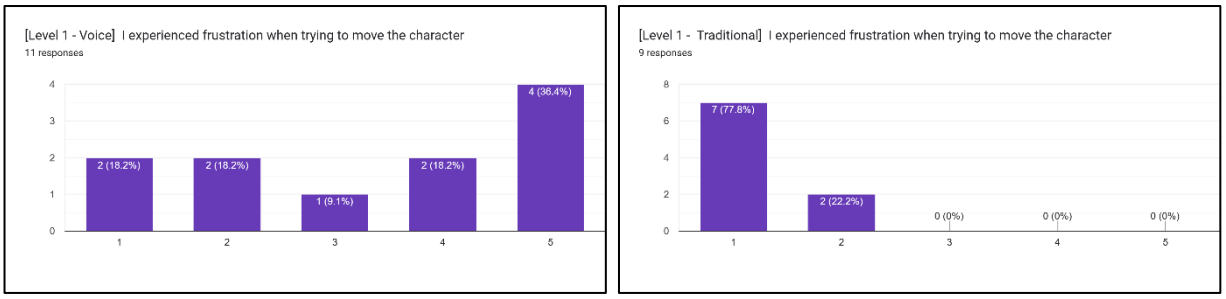

Figure 5.7 – Leve 1 frustration when moving character graphs

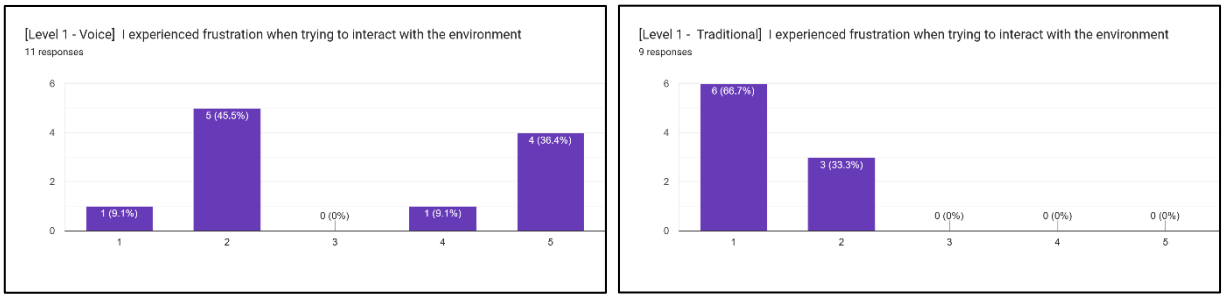

Figure 5.6 – Level 1 frustration when interacting with the environment graphs

Despite this added frustration, both groups reported comparable enjoyment (voice: M = 3.91, traditional: M = 3.56). Voice control users also reported slightly higher immersion (M = 7.00 vs. M = 6.11). This pattern of higher frustration but comparable engagement suggests that voice interaction introduced a novel challenge that participants found worthwhile in the slower-paced puzzle context.

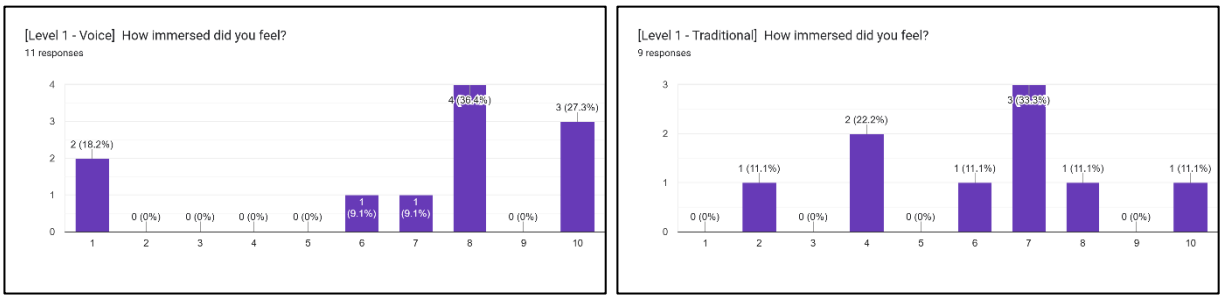

Figure 5.8 - Level 1 immersion graphs

5.4.3 Level 2

In Level 2, all 20 participants played the level twice, once with each control scheme, in randomized order. When asked which method felt more effective, 80.0% selected traditional controls and 20.0% selected voice controls ($\chi^2 = 7.20$, $p = .007$), a statistically significant preference for traditional controls.

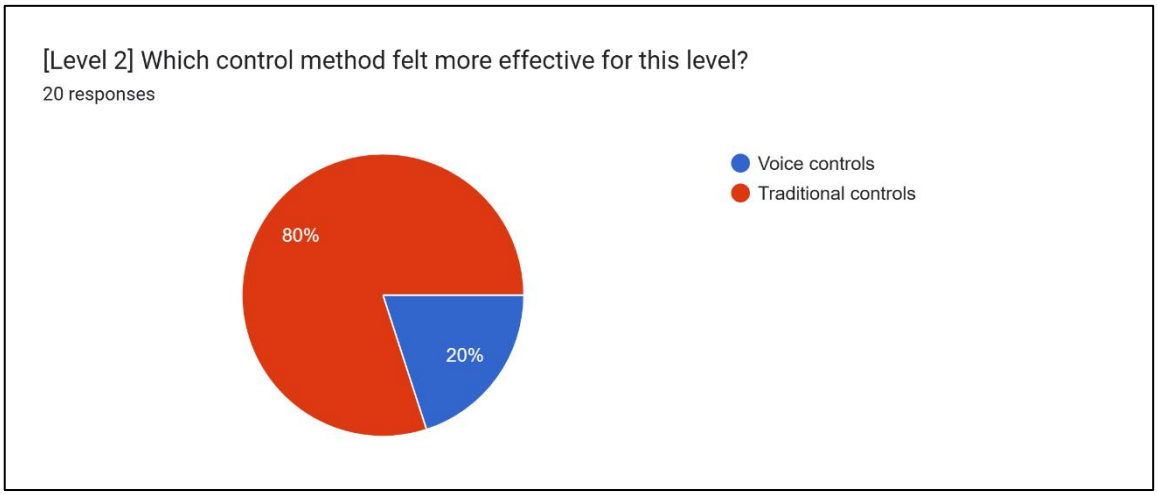

Figure 5.9 - Level 2 effective control method graph

Participants reported significantly higher frustration when reacting to the environment with voice controls (M = 3.55 vs. M = 2.45). This aligns with the qualitative feedback, where participants frequently cited the delay between issuing a voice command and seeing it executed as the main difficulty in time-sensitive situations.

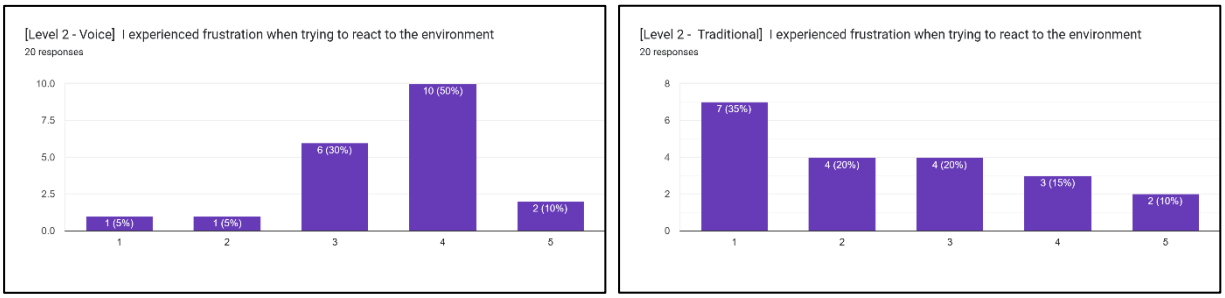

Figure 5.10 - Level 2 frustration when reacting to the environment graphs

Enjoyment was slightly lower with voice controls (M = 3.35 vs. M = 3.75). Immersion scores were close between the two conditions (voice: M = 7.15, traditional: M = 6.90), indicating that voice controls did not substantially reduce the sense of presence even in a scenario where they were perceived as less effective.

Interestingly, participants reported being more aware of the real world when using voice controls (M = 2.50 vs. M = 2.85 on the "unaware" scale). A possible explanation is that the cognitive effort of formulating spoken commands kept participants partially anchored to their surroundings, whereas traditional controls allowed more seamless absorption into the game.

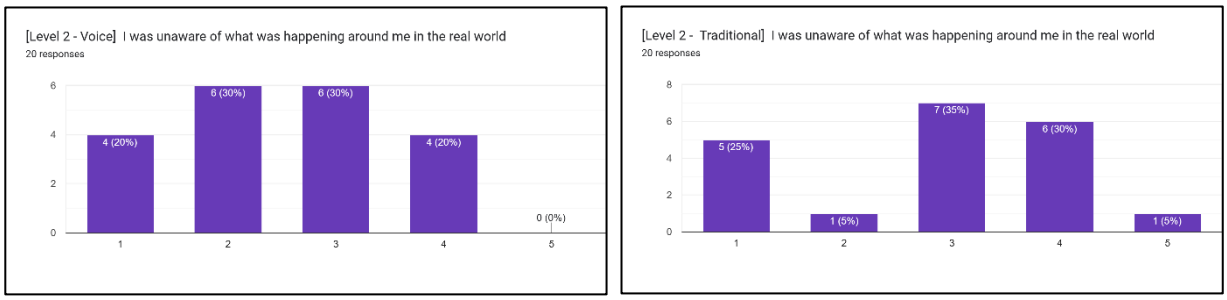

Figure 5.11 - Level 2 unaware graph

5.4.4 Level 3

Level 3 used hybrid controls exclusively, so no between-condition comparison was possible. Opinions on the intuitiveness of controls were mixed (M = 3.10, Mdn = 3.5), and enjoyment was moderately positive (M = 3.70, Mdn = 4.0). However, opinions on whether voice commands enhanced the experience were divided (M = 2.60, Mdn = 2.0), with over half of participants rating this at 2 or below.

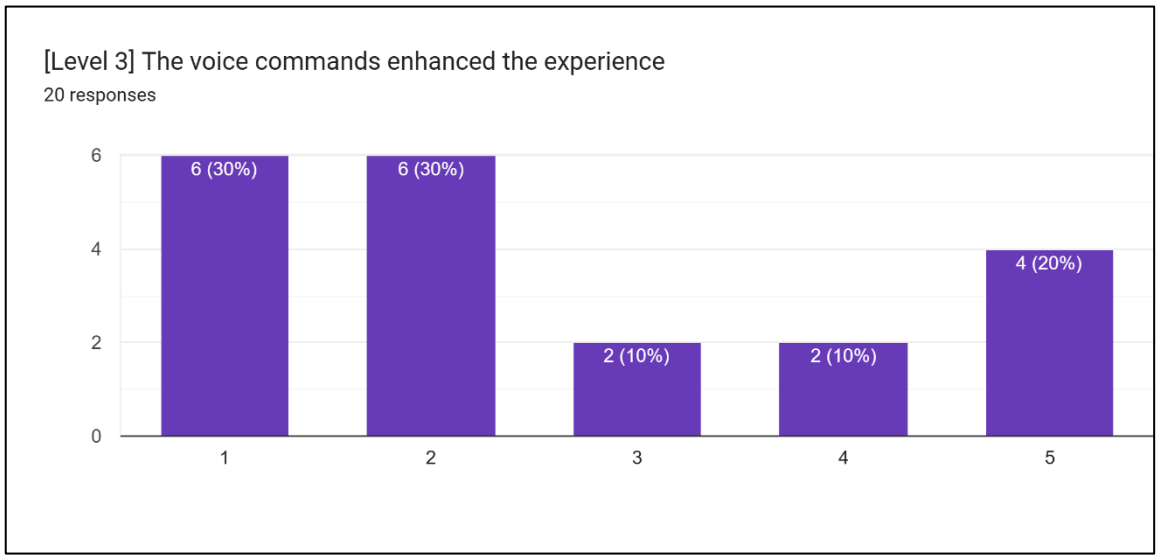

Figure 5.12 - Level 3 voice commands enhanced the experience graph

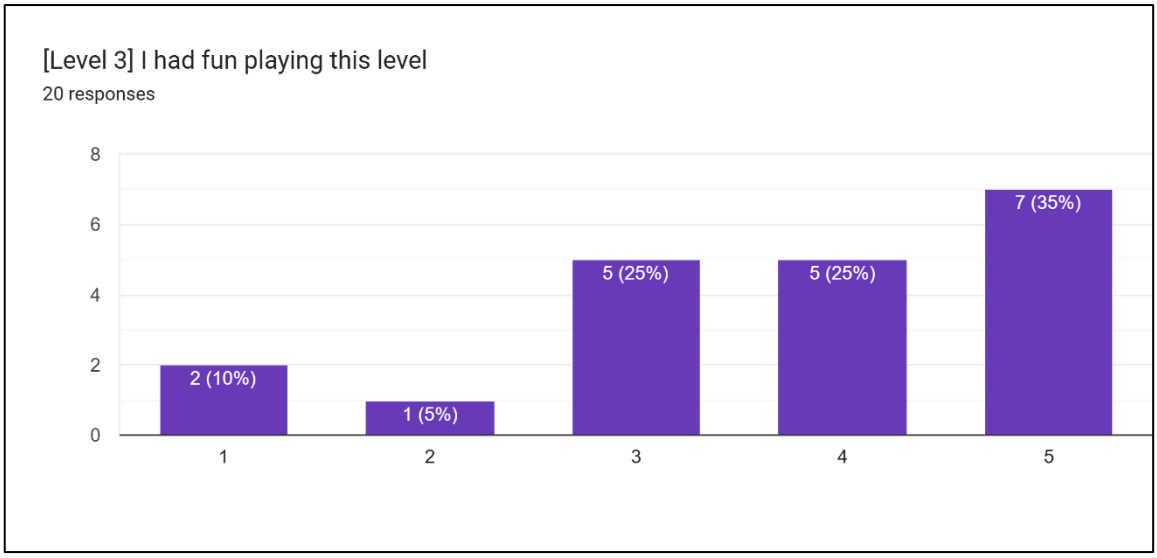

Figure 5.13 - Level 3 fun graph

Immersion averaged 6.95 out of 10. Qualitative feedback revealed that at least one participant encountered a recognition issue with the "Attack" command, which may have negatively influenced ratings for this level.

5.4.5 Overall Preferences

The most decisive finding emerged when participants identified which contexts voice controls are best suited for. Exploration (75.0%) and puzzle solving (70.0%) dominated, while fast-paced experiences received only 5.0%. The difference between puzzle solving and fast-paced was highly significant ($\chi^2 = 11.27$, $p = .001$), aligning directly with the per-level results: voice controls performed comparably in the slower puzzle scenario but introduced significant friction under time pressure.

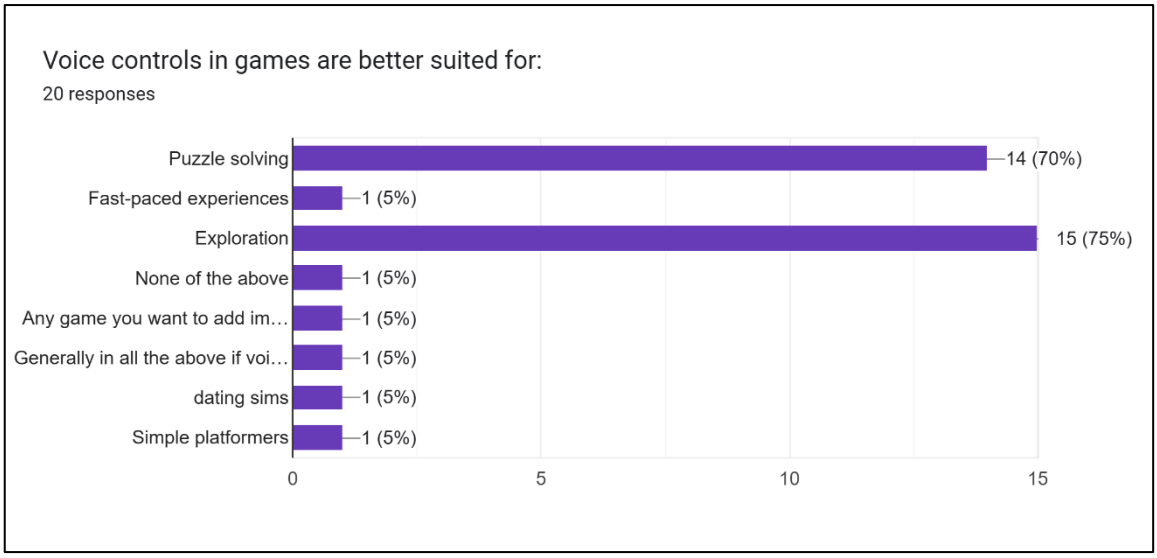

Figure 5.14 - Voice controls are better suited for graph

Chapter 6

Discussion

The results from the three levels reveal that voice controls are not universally better or worse than traditional controls but their value depends heavily on the type of gameplay they support. The following sections revisit each research question in light of the findings and consider what these results mean for the design of voice-controlled games.

6.1 Research Question 1

Do voice controls immerse the player more in the game than traditional controls?

The findings do not support the expectation that voice controls inherently produce greater immersion. In Level 1, participants using voice controls reported slightly higher immersion than those using traditional controls, while in Level 2 both conditions produced nearly identical immersion scores. When participants evaluated the three schemes overall, their preferences split almost evenly between traditional, hybrid, and voice controls, with no scheme standing out as clearly more immersive.

These results suggest that immersion depends less on the input method itself and more on how well that method fits the gameplay context. Voice controls appeared to enhance immersion in the puzzle scenario, where participants had time to formulate spoken commands and engage with the world through natural language. In the fast-paced scenario, however, the mental effort of translating intent into speech competed with the immersive experience rather than supporting it. Interestingly, participants reported greater awareness of their real-world surroundings when using voice controls in Level 2, which suggests that speaking out loud can pull players back to the physical world even while they remain engaged with the game.

6.2 Research Question 2

Do voice controls perform better or worse than traditional controls in exploration and fast-paced gameplay scenarios?

The results draw a clear line between the two gameplay contexts. In the exploration-oriented puzzle of Level 1, voice controls performed comparably to traditional controls in terms of enjoyment and understanding, even though participants reported some extra frustration when issuing movement commands. The slower pace of the puzzle let participants think about their intent, phrase it clearly, and wait for the game to respond without losing progress.

The fast-paced scenario in Level 2 produced a very different outcome. Most participants identified traditional controls as the more effective method, and frustration with voice controls rose sharply. Participants consistently pointed to the delay between speaking a command and seeing it executed, which made precise timing difficult or impossible during the platforming sections. The qualitative feedback reinforced this view, with multiple participants describing the latency and occasional misrecognition as the main obstacles to playing the level.

The final survey question, which asked participants to identify the contexts most suitable for voice controls, confirmed this pattern. A strong majority selected puzzle solving and exploration, while fast-paced experiences received almost no support. This agreement between the gameplay results and the participants' own judgments provides strong evidence that voice controls suit slower, thought-driven gameplay well but fit poorly with scenarios that demand precise timing.

6.3 Research Question 3

Does a hybrid control scheme combine the advantages of traditional and voice controls?

The hybrid scheme in Level 3 produced mixed results. Participants found the hybrid controls reasonably intuitive and reported moderate enjoyment, but their opinions split on whether voice commands actively enhanced the experience, with over half rating this aspect neutrally or negatively. This suggests that the hybrid scheme remained playable but did not fully deliver on the promise of combining the best of both input methods.

A few factors likely contributed to this result. First, the boss fight in Level 3 demanded quick timing for movement and reactions, the same context where voice controls

performed poorly in Level 2. By assigning movement to the keyboard and only combat interactions to voice, the hybrid design tried to keep voice out of time-critical actions. However, the voice-based attack and block actions still demanded quick responses, which brought back the same latency issue that frustrated participants earlier.

The Block action illustrates this tension clearly. To give players enough time to speak the command before a rock hit the character, the game displayed a short flashing indicator ahead of each falling projectile. The idea was to use the warning window to cover the delay between speaking "block" and the action executing, so that the voice command would register just in time. In practice, this timing proved difficult to tune. If the warning appeared too early, players had to wait before speaking and the threat felt artificial; if it appeared too late, the voice command did not execute in time to block the projectiles. The warning window also had to account for variable voice recognition delay, which meant that the same indicator timing could feel forgiving in one attempt and too tight in the next. A longer warning window might have made the Block action easier to use consistently, but it would have also reduced the sense of urgency that made the boss fight engaging in the first place.

The hybrid approach therefore points in a useful direction but does not solve the problem on its own. The core idea of assigning voice commands to non-time-critical actions and traditional controls to movement, holds promise, but success depends on carefully picking which actions can tolerate the delay of voice input without hurting gameplay.

Chapter 7

Conclusions

This thesis explored the role of voice interaction as a control scheme in video games and investigated how modern Large Language Models can support natural, open-ended player communication in interactive environments. Motivated by the historical limitations of predefined command grammars and recent advances in natural language processing, we set out to design, implement, and evaluate a voice-controlled game and to compare three control schemes: traditional, voice, and hybrid.

7.1 Overview

We developed a 2D platformer in Unity featuring three levels, each designed to test a specific control scheme under different gameplay conditions. The first level used puzzle-based gameplay to evaluate voice and traditional controls in a slow-paced exploratory context. The second level introduced timing-based platforming challenges to test both schemes under time pressure. The third level used a hybrid scheme in a boss battle, assigning movement to the keyboard and combat interactions to voice commands.

To translate player speech into in-game actions, we built a two-stage pipeline. The first stage transcribes speech into text using a local automatic speech recognition server based on OpenAI's Whisper, chosen for its balance of accuracy, responsiveness, and offline operation. The second stage interprets the transcribed text using Google's Gemini-2.5-flash via Firebase, applying the Reasoning via Planning framework to map natural language instructions to valid game actions. We further reinforced the pipeline with execution-level safeguards that validate each action against the current game state before execution.

We evaluated the system through a user study with 20 participants who played the game remotely and completed a post-play questionnaire. The collected data included Likert-scale responses on frustration, enjoyment, and immersion, as well as gameplay performance metrics such as completion time and failed attempts. We analyzed the results using means and medians to describe Likert-scale responses, and chi-square tests to assess categorical preferences.

The findings confirmed a clear pattern across the three research questions. Voice controls did not produce greater immersion than traditional controls overall, but they enhanced immersion specifically in the puzzle scenario, where the slower pace allowed players to engage with natural language without time pressure. In contrast, voice controls performed significantly worse than traditional controls in the fast-paced scenario, where the latency of speech recognition and command execution prevented precise timing. The hybrid scheme produced moderate results, succeeding as a playable design but failing to fully combine the strengths of both methods because some voice commands still required quick reactions. Taken together, the results indicate that voice controls work best when matched to gameplay that gives players time to think and speak, rather than when forced into contexts that demand split-second responsiveness.

7.2 Limitations

Several limitations should be considered when interpreting the findings of this study. First, the sample size of 20 participants is modest and skewed toward frequent gamers aged 18 to 34, which limits the generalizability of the results to broader audiences such as casual players, younger or older users, or players with accessibility needs, a group for whom voice controls may hold particular value. Second, most participants had no prior experience with voice controls in games, meaning their responses likely reflect a first-time learning experience rather than performance with a familiar input method. With more practice, participants might have adapted their phrasing and reduced some of the frustration observed in the fast-paced scenario.

Third, because participants completed the study remotely, gameplay conditions varied across hardware configurations, microphone quality, and ambient noise, all of which can affect speech recognition accuracy. While we required the use of an external microphone to mitigate this, we could not fully control the environment in which each session took place. Fourth, the study used a specific combination of speech recognition and LLM processing, each with its own latency and accuracy profile. Different model choices, faster hardware, or further pipeline optimization could yield meaningfully different results. Finally, technical issues such as the unrecognized "Attack" command reported by at least one participant in Level 3 directly affected the gameplay experience and may have influenced the survey responses for the hybrid scheme.

7.3 Future Work

The findings of this study open up several directions for future research. The most immediate extension would involve replicating the study with a larger and more diverse sample, including participants with varying levels of gaming experience and players who rely on voice interaction for accessibility reasons. Such a study would help determine whether the patterns observed here hold across broader populations and would clarify whether voice controls offer specific advantages for users who find traditional input methods difficult to use.

Another promising direction lies in reducing the latency of the voice pipeline. Future work could explore streaming speech recognition that begins processing audio before the player finishes speaking, lighter-weight LLMs optimized for low-latency inference, or hybrid local-cloud architectures that distribute the workload to minimize delay. As these technologies continue to improve, the range of gameplay contexts where voice controls remain viable should expand, possibly making real-time interaction feasible even in faster-paced scenarios.

The study also suggests that further investigation into hybrid control design could be done. Identifying which categories of in-game actions can tolerate the latency of voice input, and which cannot, would help designers build hybrid schemes that genuinely combine the strengths of both modalities rather than partially inheriting the weaknesses of one. Future work could systematically test different action-to-modality mappings across multiple genres to develop concrete design guidelines.

Finally, this thesis focused on using voice as a direct input method, but the integration of LLMs in games opens up broader possibilities. Voice interaction could be used not only to control characters but also to converse with non-player characters, query the game world, or shape narratives in real time. These applications would shift voice from a control mechanism to a storytelling and immersion tool, an area where the strengths of natural language are likely to be most valuable.

Bibliography

[1] F. Allison, M. Carter and M. Gibbs, "Word Play: A History of Voice Interaction in Digital Games," *Games and Culture,* pp. 91-113, 2020.

[2] X. Shuyuan, L. Zelong, M. Kai and Z. Yongfeng, "AIOS Compiler: LLM as Interpreter for Natural Language Programming and Flow Programming of AI Agents," arXiv, 2024.

[3] O. Long, W. Jeffrey, J. Xu, A. Diogo, W. Carroll, M. Pamela, Z. Chong, A. Sandhini, S. Katarina, R. Alex, S. John, H. Jacob, K. Fraser, M. Luke, S. Maddie, A. Amanda, W. Peter, P. F. Christiano, J. Leike and R. Lowe, "Training language models to follow instructions with human feedback," in *Advances in Neural Information Processing Systems*, New Orleans, 2022.

[4] Valve Corporation, "Mage Arena," Steam, 24 July 2025. [Online]. Available: https://store.steampowered.com/app/3716600/Mage_Arena/. [Accessed 2 May 2026].

[5] Valve Corporation, "YAPYAP," Steam, 3 February 2026. [Online]. Available: https://store.steampowered.com/app/3834090/YAPYAP/. [Accessed 2 May 2026].

[6] T. Kiiski, "Voice Games: The History of Voice Interaction in Digital Games," 2020. [Online]. Available: http://www.theseus.fi/handle/10024/336352. [Accessed 19 February 2026].

[7] D. Reddy, L. Erman and R. Neely, "A model and a system for machine recognition of speech," *IEEE Transactions on Audio and Electroacoustics,* vol. 21, no. 3, pp. 229-238, 1973.

[8] S. Harada, J. O. Wobbrock and J. A. Landay, "Voice Games: Investigation Into the Use of Non-speech Voice Input for Making Computer Games More Accessible," in *Human-Computer Interaction – INTERACT 2011*, Berlin, 2011.

[9] "Nintendogs," 9 March 2026. [Online]. Available: https://en.wikipedia.org/wiki/Nintendogs. [Accessed 19 March 2026].

[10] "Phoenix Wright: Ace Attorney," 13 March 2026. [Online]. Available: https://en.wikipedia.org/wiki/Phoenix_Wright:_Ace_Attorney. [Accessed 19 March 2026].

[11] "Tomb Raider (2013 video game)," 17 March 2026. [Online]. Available: https://en.wikipedia.org/wiki/Tomb_Raider_(2013_video_game). [Accessed 19 March 2026].

[12] "The Legend of Zelda: Phantom Hourglass," 8 March 2026. [Online]. Available: https://en.wikipedia.org/wiki/The_Legend_of_Zelda:_Phantom_Hourglass. [Accessed 19 March 2026].

[13] "Alien: Isolation," 11 February 2026. [Online]. Available: https://en.wikipedia.org/wiki/Alien:_Isolation. [Accessed 19 March 2026].

[14] "Tom Clancy's Splinter Cell: Blacklist," 18 February 2026. [Online]. Available: https://en.wikipedia.org/wiki/Tom_Clancy%27s_Splinter_Cell:_Blacklist. [Accessed 19 March 2026].

[15] M. Carter, F. Allison, J. Downs and M. Gibbs, "Player Identity Dissonance and Voice Interaction in Games," in *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, London, 2015.

[16] M. Biermann, E. Schweiger and M. Jentsch, "Talking To Stupid?!? Improving Voice User Interfaces," in *Mensch und Computer 2019 - Usability Professionals*, Hamburg, 2019.

[17] F. Allison, J. Newn, W. Smith, M. Carter and M. Gibbs, "Frame Analysis of Voice Interaction Gameplay," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, Glasgow, 2019.

[18] N. Zargham, J. Pfau, T. Schnackenberg and R. Malaka, "“I Didn’t Catch That, But I’ll Try My Best”: Anticipatory Error Handling in a Voice Controlled Game," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, New Orleans, 2022.

[19] A. Fraser, C. Marcus, G. Martin and S. Wally, "Design Patterns for Voice Interaction in Games," in *CHI PLAY ’18*, New York, 2018.

[20] S. Hao, Y. Gu, H. Ma, J. Hong, Z. Wang, D. Wang and Z. Hu, "Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing," in *EMNLP 2023*, Singapore, 2023.

[21] J. Park, S. Lim, J. Lee, S. Park, M. Chang, Y. Yu and S. Choi, "CLARA: Classifying and Disambiguating User Commands for Reliable Interactive Robotic Agents," *IEEE Robotics and Automation Letters,* vol. 9, no. 2, pp. 1059-1066, 2024.

[22] C. Jennett, A. L. Cox, P. Cairns, S. Dhoparee, A. Epps, T. Tijs and A. Walton, "Measuring and defining the experience of immersion in games," *International Journal of Human-Computer Studies,* vol. 66, no. 9, pp. 641-661, 2008.