Choosing the Right Voice: Evaluating OpenAI, Convai, and Inworld AI for NPC Dialogue Generation

Ashil Sabu
26397160

26397160@students.lincoln.ac.uk

School of Computer Science

College of Science

University of Lincoln

Submitted in partial fulfilment of the requirements for the
Degree of MSc Computer Science

Supervisor: Renata Ntelia

September 2023

Acknowledgements

First and foremost, I would like to extend my sincerest gratitude to my supervisor, Renata Ntelia, for her invaluable guidance, unwavering support, and mentorship throughout the course of this dissertation. Your expertise and insight have been indispensable to this project, and I am extremely grateful for the time and effort you invested in helping me. I would also like to thank my family for their constant support, encouragement, and love. To my parents, who instilled in me the value of hard work and perseverance, and my siblings, who have always been there to offer a listening ear and words of comfort—your unwavering belief in me has been a source of strength and motivation. To my friends, both old and new, thank you for your camaraderie and for the memorable experiences that we've shared. You're understanding and emotional support, particularly during the most challenging moments of this journey, have been nothing short of uplifting. This dissertation stands as a testament to the collective efforts and goodwill of all who have supported me, and for that, I am eternally grateful.

Abstract

Modern video games rely heavily on artificial intelligence (AI), particularly in the creation and behaviour of non-player characters (NPCs). NPCs are essential in producing engaging and enjoyable gaming experiences. NPCs have come to receive more attention as the gaming industry has developed, especially in RPG games. The potential for highly scripted NPCs to create a considerably more sophisticated and immersive experience has been recognised by game developers. Despite all of this, as compared to the advancements made in the development of their surrounding features, NPC interactions are still in their infancy. Modern video games' increasing complexity and interactive nature demand enhanced dialogue systems for Non-Player Characters (NPCs). These conversations add greatly to the entire gaming experience, both in terms of engagement and narrative complexity. This dissertation compares three cutting-edge AI frameworks in the context of procedurally creating NPC dialogues in games: OpenAI's GPT model, Convai, and Inworld AI.

This study takes a multi-dimensional evaluation strategy, taking key performance factors such as dialogue quality, contextual relevance, and real-time responsiveness (latency) into account. The findings provide essential insights into each AI framework's strengths and flaws, shining light on their practical consequences in game development. While OpenAI's GPT models excel in generating linguistically rich dialogues, Convai outperforms in real-time. Inworld AI, on the other hand, has exceptional contextual awareness, making it excellent for games with complex stories. This comparative study seeks to be a starting point for game creators and researchers to make educated judgements about including AI-driven dialogues into future gaming experiences.

Contents

Acknowledgements ii

Abstract iii

1 Introduction 1

1.1 Aims, Objectives and Hypothesis 1

Objective 1

Hypothesis 2

1.2 Background 2

1.3 Report Structure 3

2 Literature Review 4

2.1 AI in Gaming: A Historical Perspective 5

2.2 Non-Player Characters (NPC) 6

2.3 Procedurally Generated Languages for Non-playable Characters 6

2.4 Conversational AI for NPCs 7

2.5 Natural Language Processing 8

2.6 Ethical Issues in Automatic Dialogue Generation 8

2.7 Conclusion 9

3 Methodology 10

3.1 Project Management 10

3.1.1 Guiding Principles 10

3.1.2 Project Management Methods 13

3.2 Software Development Methods 14

3.3 Research Methods 17

3.3.1 Experimental Design 17

3.3.2 Research Methods Selection 18

3.3.3 Experimental Procedure 19

3.3.4 Statistics 21

3.3.5 Requirements 23

3.3.6 Software Design Documentation 25

4 Software Development 28

4.1 Toolsets and Machine Environments 28

4.1.1 Game Engine: Unity 28

4.1.2 Platform: PC (Windows/Mac/Linux) 29

4.1.3 Programming Language: C# 31

4.1.4 Version Control: Git 33

4.2 Implementation 34

Step 1: Collecting the APIs 34
Step 2: Creating the Unity Scene 35
Step 3: Creating the NPC characters 36
Step 4: Integrating the AI Framework APIs 37
4.3 Testing 43
5 Results 47
5.1.1 Software Solution Results 47
6 Discussion and Evaluation 51
6.1.1 Quantitative Content Analysis(CQC) 51
6.1.2 Performance Analysis 53
6.1.3 Contextual Analysis 54
6.2 Evaluation and Reflection 56
7 Conclusions 58
7.1 Future Work 59
• References 60

1 Introduction

The video game industry is undergoing remarkable technological progress, including breakthroughs in visuals, physics, and user interfaces. However, one aspect of this progression that is frequently ignored is the quality of NPC (Non-Player Character) conversations. Dialogues are necessary not only for progressing the game's plot, but also for increasing user engagement and immersion. Given the rise of AI and its growing significance in natural language creation, AI-based solutions for NPC interactions have become a significant focus for both researchers and developers. There are a lot of frameworks available for this application and this project aims to compare three of the popular frameworks OpenAI's GPT models, Convai, and Inworld AI, each with its own set of advantages and disadvantages. The extant literature, however, mainly examines these technologies independently, frequently in non-gaming situations such as customer service bots or generic conversational agents. A noteworthy gap in the present research is a targeted, comparative investigation of video game NPC dialogues. By comparing OpenAI, Convai, and Inworld AI, this research aims to offer a holistic view of the current capabilities of AI in enhancing interactive dialogues in video games, thus serving as a foundational resource for both academia and industry.

1.1 Aims, Objectives and Hypothesis

The primary aim of this dissertation is to present a complete evaluation of three popular AI frameworks in the context of procedurally generating dialogues for Non-Player Characters (NPCs) in video games: OpenAI's GPT models, Convai, and Inworld AI. The study's goal is to determine which of these technologies is the most effective in terms of dialogue quality, real-time responsiveness, and contextual relevance inside a gaming environment.

Objective

1. Evaluate Dialogue Quality: To evaluate the language richness, coherence, and level of involvement of dialogues generated by each AI framework.
2. Measure Real-time Responsiveness: To calculate the latency involved in dialogue creation for each framework and the influence on user experience.
3. Analyse Contextual Relevance: To assess how well each AI model comprehends and responds to the game's background, narratives, and player interactions.

Hypothesis

H1: OpenAI's GPT models will excel at generating dialogues with high linguistic richness and coherence, but real-time responsiveness may suffer.

H2: Convai will deliver the most efficient real-time performance, but it may fall short in terms of generating contextually rich interactions.

H3: Inworld AI will have the highest level of contextual awareness and relevance, especially in games with complicated stories.

By meeting these objectives and testing these hypotheses, the study seeks to provide a thorough comparative analysis that can guide future research and practical implementations in the fast-developing field of AI-generated NPC dialogues in video games.

1.2 Background

Over the last decade, the video game industry has seen extraordinary technological improvement, particularly in increasing the richness and interactivity of in-game worlds. One of the most important components of this interactivity is the interaction with Non-Player Characters (NPCs). Historically, dialogues were hardcoded into the game, providing limited alternatives and severely limiting player agency. The current incorporation of Artificial Intelligence (AI) technologies into gaming environments, on the other hand, has opened new options for improving NPC talks, making them more dynamic, context-aware, and engaging. Several AI frameworks have emerged as frontrunners in the field of natural language production for video games in the present landscape. Among these, OpenAI's GPT models, Convai, and Inworld AI stand out, each with its own set of strengths and challenges. OpenAI's GPT models are well-known for their language abilities, but there are still concerns regarding their real-time reactivity in interactive environments. Convai, on the other hand, is designed for efficiency, but its potential for generating contextually rich discussions has yet to be completely explored. Inworld AI promises great contextual awareness, which is especially useful for games with complex narratives, although it may necessitate more computational resources.

Existing literature assesses these frameworks in isolation or in various application domains such as customer service or conversational agents. A complete, head-to-head assessment of their performance and usefulness in the context of video game NPC dialogues, on the other hand, is significantly lacking. This study tries to close this gap by comparing OpenAI's GPT models, Convai, and Inworld AI in procedurally creating NPC dialogues across several video game genres. It aims to assess these frameworks on a variety of criteria, including dialogue quality, real-time responsiveness, contextual relevance, computational efficiency, and user experience. By doing so, this study hopes to provide the groundwork for future research and practical applications in the fast-developing field of AI-integrated video games.

1.3 Report Structure

This report is divided into 7 parts:

- Introduction: Provides a summary of the research project's goals, objectives, and hypothesis.
- Literature review: A comprehensive review of related academic works.
- Methodology: An overview of project management strategies, software development procedures, and research techniques.
- Software development: A synopsis of the proposed system's development cycle.
- Results: This section presents the findings from the research effort.
- Discussion and Evaluation: This section explores the findings and provides a critical assessment of them.
- Conclusion: This part summarises the study project's findings and any takeaways, as well as future work.

2 Literature Review

A continuous pursuit of realism, immersion, and narrative depth has transformed the world of video game creation dramatically. Non-Playable Characters (NPCs) are becoming more crucial in determining how the player's experience is shaped in this dynamic change. These digital entities are now growing into complex, interactive individuals that can engage players in rich, dynamic interactions. Previously, they were limited to written dialogue and predefined behaviours. The incorporation of Conversational Artificial Intelligence (Conversational AI) into the very foundation of digital worlds is what is driving this transition.

Natural language processing and artificial intelligence are combined to create conversational AI, which gives computers the ability to understand, produce, and reply to human language. It functions by splitting phrases into their fundamental components, accommodating the many peculiarities of human language, and recognizing that there is data or a command that must be processed. The development in this new field is pushing the boundaries for synthesizing "human-esque" dialogues, from context management to sentiment analysis and language support. Conversational AI has already started to change the rules and establish new ones, from voice assistants that can provide context-based real-time responses to automated language translators that can accurately translate a wide range of languages according to the intent of input. Said to be the logical successor of the chat-bot, it has the potential to not only improve but also potentially redefine NPC interactions in the context of gaming. The diversity of AI capabilities opens the possibility for NPCs to interact with players in immersive, adaptive, and contextually relevant ways, drastically altering how players perceive the stories and settings of video games.

This literature review explores the applicability of Conversational AI for NPCs in contemporary video games while taking a tour through the worlds of gaming and AI. In this investigation, we examine the methods that support the incorporation of Conversational AI into game design. We begin with a brief overview of artificial intelligence (AI) in gaming and how it has changed the sector over time. Further, we move our attention to NPCs and examine how they have evolved through various strategies, whether it be a straightforward improvement in their approachability or cutting-edge techniques that have completely changed their function in open-world games. Furthermore, we try to identify the ethical issues involved in the automatic generation of NPC statements in digital games.

2.1 AI in Gaming: A Historical Perspective

Since the beginning of the medium, there has been a continual effort to incorporate artificial intelligence into games. In the beginning, AI was mainly used to enable simple opponent behaviours in games like chess or early arcade games. These primitive AI systems provided predictable and frequently rigid gameplay experiences by using rule-based decision trees. However, as device capabilities increased, gaming AI complexity increased as well. Pathfinding algorithms' development in the 1980s made it possible for NPCs to move across virtual environments with more intelligence, giving the impression of autonomy. NPCs that roamed the globe and provided players with hints and interactions that were previously unusual were first introduced in games like "The Legend of Zelda" (1986). NPCs started to become more than just movable quest givers after this.

AI in games is generally driven by non-playable characters or NPCs and the behaviour of NPCs is driven by AI (Togelius & Yannakakis, 2016). Everything that creates the appearance of intelligence is termed game AI, which should increase the game's difficulty, immersion, and enjoyment. Deterministic and nondeterministic approaches are typically used in game AI (Drageset, et al., 2019). Deterministic AI is composed of predetermined, hard-coded AI programmes. A straightforward example would be "follow the player," however by combining Finite State Machines with Behaviour Trees, it is possible to create more complicated and difficult AI. The opponent can adapt to the approaches based on what has already happened, and machine learning algorithms are used to get stronger each time to satisfy the gamer's appetite. Non-deterministic AI points to ambiguity in decision-making (Perez-Liebana, et al., 2019).

Game creation saw a major shift when decision trees and finite state machines were adopted as more advanced AI algorithms (Rabin, 2006). These methods, gave NPCs the ability to act in a semi-autonomous manner, boosting player-NPC interactions in limited settings. A genetic algorithm is a method that simulates biological evolution. The NPC can select from a set of procedures, algorithms, and parameters. These methods, which are based on natural evolution, are effective in determining the global or near-global optimal solution to optimisation issues (Anon., 2011).

2.2 Non-Player Characters (NPC)

Non-player characters (NPCs) are a common feature in contemporary videogames, particularly role-playing games (RPGs). Evidence suggests player relationships with these fictional, digital characters can manifest as deeply emotional experiences that can ‘bleed’ off the screen and affect the daily lives of players (Scriven, 2023). An integral part of social believability in role playing games is believability of non-player characters (NPC) (Pickett, et al., 2021). Creating non-player game characters (NPCs) with human-like behaviour is one of the most ignored issues in artificial intelligence (AI) for computer games. Modern NPCs use specific decision-making processes to determine their actions in various scenarios, allowing them to affect the present state of the game environment. Sir Henrik Waperfelt also stated in his 2016 PhD thesis (Warpefelt, 2016) that the video game business has experienced a tremendous change in terms of technological improvement and the adoption of newer technology such as Artificial Intelligence. However, one area of game creation that went unnoticed and struggled to keep up with the trend was NPC development. NPCs are used not just in games, but also in many simulations where a character must be placed in a virtual world and is not controlled by any user (Suyikno & Setiawan, 2019). Previously, games using NPCs relied on a script. The NPCs were not dynamic and were unable to respond to unpredictable player behaviour, resulting in decreased player involvement and immersion in gaming. NPCs rely on established algorithms or a complex AI technique to generate the needed lifelike player agents (Agis, et al., 2020).

The most prevalent technique of introducing NPCs is using Behavioural Trees (BTs). When it comes to NPC behaviour creation, Behaviour Trees outperform Fine State Machines (FSM). This is because of aspects like as reusability, ease of design, scalability, and modularity (González-Sánchez & Vela, 2014). Every NPC action in a behaviour tree is based on a hierarchy (Caroux, et al., 2015). BTs are also tried and tested, having been employed in popular games such as Halo, Far Cry: Primal, and Tom Clancy's The Division.

2.3 Procedurally Generated Languages for Non-playable Characters

Simulation games like “Spore” (2008) and “Black and White” (2001) as well as some adventure games “No Man’s Sky” (2016) and “Mass Effect” (2007) include the procedural generation of agent populations. These agents should be fascinating, surprising, and adaptable to their surroundings. That is, these agents' physical and behavioural features should evolve because of their physical and social environments, as this adds consistency and realism to the simulation while improving the player's impression that their decisions are genuinely meaningful. Recent research has shown that it is possible to train agents to create language from scratch to win a referential game (Lazaridou, et al., 2018). The agents used both human-annotated and pixel data to play the referential game. The agents learnt via deep reinforcement learning and have handmade architectures from the start (Rawal & Miikkulainen, 2018). Emergent communication has been used to facilitate agent cooperation in multi-agent environments (Smith, et al., 2020). During training, these agents were given more information about other agents in their surroundings and had fixed, handcrafted architecture.

2.4 Conversational AI for NPCs

Conversational AI is a subset of Artificial Intelligence that focuses on using machine learning to create applications that enable human-like conversations with machines. It simulates and automates conversational exchanges using speech and text-based agents, with the goal of providing a realistic chatting experience. Conversational AI allows humans to use natural language to speak with an automated system. The interaction could be verbal or written. It may be sent to the user via chat channels (for example, Facebook Messenger and Skype), specific phone or web applications, incorporated into a website, or deployed as part of an operating system (Ruane, et al., 2019). Industry titans such as Apple, Microsoft and Google have developed Intelligent Voice AI assistants by harnessing the powers of conversational AI (Tian, et al., 2020).

In terms of practical integration, there are several language models that have been pre-trained to some extent that are publicly available. As the surrounding fields for this technology have advanced, so have the size and quantity of parameters of these models. Conversational AI models like GPT-3 have been employed to create more engaging and interactive NPCs (Brown et al., 2020). Chatbot technology has also been integrated into games, allowing for a wide range of conversational outcomes based on player input (Li et al., 2021). Models trained on public datasets that are widely available produce poor results, owing to a lack of context for the specific application for which they are being trained. As a result, models such as BERT are being fine-tuned to better suit the application in which they are employed (Mehta, et al., 2021).

2.5 Natural Language Processing

The community has a shared interest in applying natural language processing (NLP) tools to manage and moderate believable and contextual interactions in modern games. In comparison to the work done in managed scripted systems, there is less study in the field of autonomous natural language interfaces (Davies, et al., 2021). Natural language generation has been aided by generative pre-training across a variety of tasks. This method, known as GPT-2, builds a language model from many unlabelled text corpora and then fine-tunes it to a specific context using a transfer learning approach (Radford, et al., 2018). This has since been extended to GPT-3, where 175B parameters are utilised in the language model and it has been demonstrated that it can generate text that human judges have dithered about (Brown, et al., 2020).

2.6 Ethical Issues in Automatic Dialogue Generation

Dialogue systems' abilities can be divided into two categories: task-oriented dialogue and non-task-oriented dialogue. Models with this capability are also known as "chatbots" or "chit-chat models." Both abilities are required in digital gaming. An NPC that informs the player of the next goal or assignment, for example, necessitates task-oriented conversational abilities. Based on data from "World of Warcraft," (Stegeren & Myśliwiec, 2021) tried to synthesise quest material and NPC voice (quest description) from the quest title. When the player wants to have a casual discussion with a character in the game environment, however, the NPC must be able to communicate in a non-task driven manner. Personas in digital games are meant to alter as the game continues. The novel frequently portrays the process through which persons grow both personally and mentally. NPCs may also change because of player interaction.

Bias is described as prejudice for or against a person, group, idea, or thing, especially when expressed unfairly. prejudice is a broad phrase that encompasses a wide range of issues that are particularly relevant in natural language systems, such as proclivity for regional speech patterns, discriminatory gender prejudice, personal point-of-view bias, and countless more. These prejudices can be conveyed discreetly (for example, through verbal signals indicating a point of view) or openly (for example, through discriminating words against a gender). Dialogue models are vulnerable to unconsciously encoding inherent biases in human dialogue due to their subjective nature and purpose of replicating human behaviour (Henderson, et al., 2017).

2.7 Conclusion

This literature survey provides a complete overview of the quickly changing landscape of artificial intelligence in video game production, with a focus on Non-Playable Characters (NPCs). This review of the literature has provided important insights into the evolution of AI in games, notably the role and complexities of NPCs. Early gaming AI was simple, but it has evolved to include complicated algorithms such as Behavioural Trees. NPCs have progressed from static quest-givers to emotionally compelling people, and the introduction of procedural languages and Conversational AI technologies like as GPT-3 has made them more dynamic and interactive than ever before. The literature review highlights the ethical dimensions of automatically producing dialogue for NPCs, a topic that becomes increasingly important as these characters get more nuanced and lifelike. Bias, representation, and the potential for stereotype reinforcement are all issues that researchers and developers must be aware of. Based on these findings, this research will perform experimental comparisons of OpenAI, Convai, and Inworld AI for procedurally creating NPC dialogues. The naturalness of communication, contextual awareness, ethical factors such as bias, and computational efficiency will all be evaluated. To enable these evaluations, a software solution will be developed with the goal of giving evidence-based suggestions for incorporating Conversational AI in video games. In conclusion, the literature analysis lays the groundwork for the project by identifying crucial areas of attention in the fast-growing convergence of AI and game development.

3 Methodology

3.1 Project Management

Managing a research project incorporating different AI frameworks for NPC dialogue creation presents unique issues that necessitate strategic planning, resource allocation, and risk avoidance. This section describes the project management approaches that were developed to address the needs of this specific research project.

3.1.1 Guiding Principles

The management of this project will be guided by several principles, each selected to ensure that the project not only meets its research objectives but also adheres to best practices in project management.

1. Iterative Development: Given the fluid nature of AI technology, an iterative approach allows for incremental progress and adaptability (Conforto, et al., 2014). The development of this project will be in small iterations where a usable product will be developed at the end of each iteration and will be refined in the following iterations.
2. Quality Assurance: Quality assessments will be done at each level to ensure that the research findings are reliable and replicable (Larson & Gray, 2011). Quality metrices applicable for this research project will be considered and evaluated in each iteration of the development to ensure the quality of the developed product.
3. Risk Management: Proactive risk identification and mitigation will be continuous throughout the project. When identifying risks, one must analyse what might happen, why it might happen, and which risks threaten the project's success (AlbadarnehIsraa, et al., 2015).

Following are the main tasks that are pivotal for this project development:

1. Planning Phase:
    - Literature Review: Collect and analyse existing academic papers, case studies, and other relevant materials to set the groundwork for the research.
    - Identify Research Methods: Decide on the research methodology, whether qualitative, quantitative, or mixed.

- Prototype Design: Define the requirements and specifications for the project prototype.
- Tool and Platform Selection: Choose which platforms (OpenAI, Convai, Inworld AI) will be the focus of the study.

2. Implementation Phase:
    - Prototype Development: Develop the software prototype and integrate AI components for NPCs.
    - Data Collection: Gather data regarding the quality, latency, and other metrics of NPC dialogues generated by each AI platform.
    - Experimentation and Testing: Run experiments to test each platform's capabilities.
3. Analysis Phase:
    - Data Analysis: Analyse the data collected from the testing.
    - Result Interpretation: Make sense of the data and draw conclusions regarding the effectiveness of each platform in NPC dialogue generation.

By approaching each of these major activities methodically, the research project can be conceived, implemented, and completed with useful insights and important contributions to the area.

To establish the period for each main task and subtask, a Gantt chart will be used. This aids in tracking progress and makes managing dependencies between activities easier. Gantt charts provide a visual representation of a project's timeline, breaking it down into main tasks and subtasks and determining the time allotted for each (Kerzner, 2022). The following Grantt Chart demonstrates the timeframe for each of these tasks.

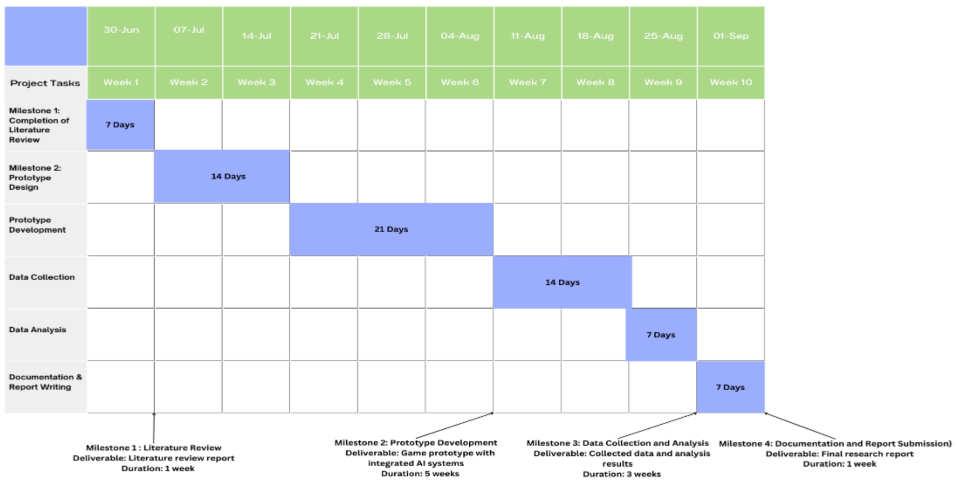


Figure 3.1: Grantt Chart demonstrating the timeframe for each of the tasks.

There are various risks associated with this project. Some of them are:

- Inconsistency in data: The quality of NPC talks created by AI platforms may fluctuate, impacting the research's validity.
- Resource Constraints: Limited computational resources may make it difficult to conduct extensive tests on each AI platform.
- Ethical Issues: The usage of AI-generated dialogues may bring ethical issues that must be addressed, potentially delaying the project.
- Timeline risks: Delays in any of the primary tasks could cause the project's completion date to be pushed back. The project's development period was limited to 45 days, and the project presented numerous technological obstacles, making it difficult to complete on time.
- Financial challenges: Because the project did not have any funding, it was critical for the developer to keep expenses to a minimal. This means that all the tools and platforms used for the development of this project had to be free and open source.

The risks associated with this project can be identified and mitigated using SWIFT (Structured What-If Technique). SWIFT is a structured brainstorming method for risk identification, commonly used for its simplicity and effectiveness (Holtz, et al., 2023). SWIFT sessions can be held where unforeseen risks can be identified, and ideas can be brainstormed to mitigate the risks.

Given that the research involves code for testing AI algorithms, data sets, and multiple drafts of the dissertation, a version control system like Git will be indispensable. The project seeks to fulfil its objectives efficiently within the given schedule by carefully planning and managing these special needs. Each of these project management tools and tasks is designed to assist the research's core goals and objectives.

3.1.2 Project Management Methods

The primary goal of this project is to compare different AI algorithms for producing NPC speeches. As a result, combining traditional and agile project management strategies might be advantageous. Following are some project management strategies that may be relevant to this project:

1. Agile Project Management (APM): Because AI and machine learning research is experimental and dynamic, Agile methodologies allow for iterative cycles and flexibility in the research process (Denning, 2020). The iterative nature of agile processes also allows the developers to uncover and address issues early on, for example, if a model underperforms or if the quality is lower than expected (Vial, et al., 2022). The major advantage of using Agile project method in this research is the iterative development life cycles and the flexibility. By focusing on iterative delivery, Agile allows for early discovery of issues or changes, making it easier to adapt and reducing the impact of risks (Kerzner, 2018).
2. Scrum: Scrum, as a component of Agile frameworks, can be employed for its sprint-based methodology, which aids in keeping a steady work pace and allowing for quick adjustments. Scrum is particularly well-suited for the project on NPC dialogue generation. Sprints can be dedicated to coding, testing, and refining individual algorithms. It's very beneficial for coding and testing AI algorithms (Rubin, 2012). The sprint approach in scrum enables the early delivery of critical features, lowering the risk of late failure and allowing for early course correction (ALI, et al., 2023).
3. Kanban: Kanban is an Agile methodology that emphasizes visualizing work, limiting work in progress, and delivering work in a continuous flow. Kanban projects are managed through a Kanban Board, which is a visual representation of the work being done. The Kanban Board is divided into columns, each of which represents a stage of the project. As work is completed, each activity is moved to the next column on the board (Uzwyshyn, 2023). Used for task and workflow

tracking, Kanban enables for visual management of project operations, making it easier to spot bottlenecks and keep tasks flowing smoothly (Ahmad, et al., 2017).

Employing these project management methods comes with some disadvantages as well. These methods have the following disadvantages to the project:

1. Agile Project Management (APM): Agile-only projects may suffer from a lack of documentation and a perceived lack of control, which can lead to scope creep. Also, constant iterations can be time-consuming and may detract from the project’s main objectives (Conforto, et al., 2014). To solve the drawbacks, the project can combine Agile and traditional approaches (commonly referred to as 'Water-Scrum-Fall') to add structure and governance without sacrificing flexibility (Reiff & Schlegel, 2022).
2. Scrum: Scrum's flexibility can occasionally lead to scope creep, in which things are added without adequate thought for the impact on the overall project (Gandomani & Ziaei, n.d.). Strict backlog prioritization should be enforced to mitigate this.
3. Kanban: Kanban does not emphasise time-based goals, which can lead to delays if not managed appropriately (Ahmad, et al., 2017). Developer must be diligent in pulling work and updating the Kanban board given the freedom and flexibility (Anderson, 2010).

3.2 Software Development Methods

A Software Development Method, also known as a Software Development Methodology, is a framework for structuring, planning, and controlling the software development process. To guide the many stages of software development, these methodologies include diverse practises, techniques, and procedures. Tasks such as requirements analysis, system design, coding, testing, and maintenance are frequently included (Sommerville, 2015).

The selection of a particular software development approach in this project will be influenced by several factors, including but not limited to:

- Dynamic Content Generation: The project requires real-time, adaptive NPC dialogues based on various player interactions. Unlike static content, which remains constant for every user, dynamic content changes according to various factors such as context, user preferences, and environmental variables.

- User Experience: User Experience (UX) is a broad term that refers to the total experience and satisfaction a user has while interacting with a product or system. In the context of NPC (Non-Player Character) dialogue systems in video games, UX is crucial for creating an immersive and emotionally engaging environment. The quality of the dialogues, their relevance to the story, and their adaptability to player actions all play a pivotal role in the overall UX. The dialogues should be natural, contextually relevant, and engaging to enhance player immersion.
- Scalability and Adaptability: Scalability refers to the system's ability to handle a growing amount of work or its potential to be enlarged to accommodate that growth. Adaptability is the system's capacity to be modified for different conditions or use cases. Both are critical factors in designing NPC (Non-Player Character) dialogue systems that not only serve current needs but also can evolve with technological advancements and varying user demands. As the gaming environment evolves, the dialogue system must be able to scale and adapt without requiring comprehensive code changes.

For the development of this project, various software development methods were considered. Some of them are:

1. Waterfall Model: The Waterfall model is a traditional model for software development that emphasizes a logical progression of steps across major phases such as Requirements, Design, Implementation, Testing, Deployment, and Maintenance. Because it is linear and phase-dependent, the Waterfall model is unsuitable for projects that require agility and adaptability (Sommerville, 2015). The Waterfall model has long been critiqued for its 'all or nothing' approach, particularly in projects with high degrees of uncertainty (Khan, 2023). Also, the sequential phases in the Waterfall model often led to longer project timelines. Testing comes at the end of the Waterfall Model. This could cause crucial insights to be lost in a project that requires iterative evaluations of several AI frameworks to influence future actions. Also, the Waterfall Model frequently mandates that each phase be finished before proceeding to the next, resulting in potential

downtime between stages and wasteful resource use, which is not suitable for time-sensitive academic projects like this (Cooper, 2016).

2. Agile (Scrum): Agile methodologies, particularly Scrum, offer iterative development that can adapt to the evolving needs of NPC dialogues (McKenna, 2016). An Agile framework like Scrum may be a lot more appropriate strategy than traditional methods such as the Waterfall model in a research-intensive and dynamic undertaking like comparing AI frameworks for generating procedural NPC dialogues. Scrum's iterative structure allows to create minimum viable products (MVPs) in short cycles, which is ideal for technology evaluation projects (Hussain, 2020).
3. DevOps: With an emphasis on automation and continuous integration, DevOps may be advantageous but may be excessive for a project with such a narrow scope (Kim, et al., 2021). Typical DevOps practises include, among other things, continuous integration/continuous deployment (CI/CD), automated testing, and infrastructure as code. These practises are primarily useful for large-scale, long-term initiatives that necessitate a high level of collaboration, rapid deployment, and operational dependability (L’Esteve, 2023). DevOps focuses not just on development but also on the software's operational elements. Given that the project is research-focused and does not require the maintenance of a live service, the operational benefits of DevOps are superfluous (Kim, et al., 2021).
4. Kanban: For a project with a more flexible schedule and tasks that don’t fit neatly into sprints, Kanban can offer the adaptability needed (Kumar, et al., 2022). Kanban gives a visual representation of the workflow, making it easier for individual researchers or small teams to stay on top of project status. This is especially beneficial when evaluating multiple AI frameworks at the same time (Ahmad, et al., 2013). Kanban, as opposed to rigid techniques such as Waterfall, allows for adjustments even late in the development process. This versatility can accommodate the project's increasing research requirements. Given one of the key evaluation criteria is real-time responsiveness, Kanban's focus on immediate and continuous improvement can offer a methodological parallel to what you aim to measure in AI frameworks.

Given the project's needs for adaptability, iterative feedback, and scalability, a combination of Scrum and Kanban is chosen for structured development and Kanban for flexibility. Combining Scrum with Kanban—also known as "Scrumban"—can provide a balanced strategy that combines the organised development cycles of Scrum with the real-time, flexible nature of Kanban. This combination is especially appropriate for this research, which aims to evaluate several AI frameworks for NPC interactions in video games based on criteria such as dialogue quality, real-time responsiveness, and contextual relevance. Scrumban enables the developer to adapt to changing requirements, which is critical given the nature of this project's study. This ensures that new insights or frameworks can be quickly integrated into the research (Hidalgo∗, 2019). Structured sprints in Scrum give defined timeframes for monitoring work and receiving feedback. These feedback loops are crucial for academic undertakings that rely on scholarly review (Serrador & Pinto, 2015). While Scrum provides a framework for expanding the project by adding new features, Kanban ensures that the expansion is efficient by visualising bottlenecks in real-time. Scrum and Kanban both emphasise reflection and adaptation, which aligns with the ethos of academic research, which is frequently about constantly refining and expanding on current knowledge. Given the project's complexity and time constraints, a combined Scrumban method provides the essential foundation and flexibility for efficiently managing multiple components, making it an appropriate solution.

3.3 Research Methods

The aim of this research is to assess the effectiveness of three AI frameworks—OpenAI's GPT models, Convai, and Inworld AI—in creating dialogues for NPCs in video games based on dialogue quality, real-time responsiveness, and contextual relevance. A multi-method research approach combining both qualitative and quantitative methods is proposed to offer a comprehensive analysis.

3.3.1 Experimental Design

A multi-method research methodology is employed to successfully address the research questions and objectives. This method enables a thorough assessment of each AI framework in the context of procedurally creating dialogues for NPCs in video games. The various strategies are appropriate for the complexity of the objectives. The combination of qualitative and quantitative methodologies provides for a thorough knowledge of the AI frameworks. While quantitative tools provide measurable and comparative indicators, qualitative methods provide more nuanced insights regarding dialogue quality and contextual significance. By providing alternative viewpoints on the same subject, the use of multiple approaches improves the rigour of your research. This is critical in the ever-changing field of artificial intelligence, where a comprehensive approach might result in improved generalizability. Furthermore, by employing various methodologies, we can evaluate and cross-reference the results to guarantee that they are reliable and not the consequence of methodological bias or restrictions. The concept of validity in research refers to the extent to which an instrument measures what it is designed to measure. Using many approaches allows for cross-validation of findings, boosting the overall validity of the research.

3.3.2 Research Methods Selection

The following research methods can be employed to address the research questions:

1. Quantitative Content Analysis: A Quantitative Content Analysis (QCA) is used to evaluate dialogue quality based on preset measures such as linguistic richness, coherence, and level of involvement. Quantitative Content Analysis is a research method that is often used to quantify specific properties in a set of data in a systematic and objective manner. This enables a formal and objective comparison of dialogue quality across different AI frameworks (Neuendorf & Kumar, 2016). This method aids in making the different AI frameworks directly comparable in terms of dialogue quality by translating qualitative text data into measurable measurements. In this situation, it can be a very useful tool for assessing the quality of AI-generated dialogues for NPCs in video games. Quantitative content analysis provides objective measures to form a more rounded view. However, the environment in which dialogues are created is sometimes lost in quantitative content analysis. Furthermore, this review can be time-consuming and may necessitate many rounds of testing for dependability checks.
2. Performance Testing: This method can be employed to measure the real time responsiveness of the models. Performance testing provides accurate measures for latency involved in dialogue creation for each AI framework. For statistical rigour, this can be measured in milliseconds and documented across several trials (Whiting & Datta, 2021). We can assess how each framework will scale in a real-world game environment by testing it under varied loads. However, many times, performance testing takes time and may necessitate large computational resources.

3. Contextual Analysis / Case Study: To analyse the contextual relevance of the generated dialogues, each AI framework can be analysed in a simulated game environment to see how effectively it understands and responds to various scenarios. Relevance and appropriateness of each dialogue in the context of the game can then be documented. This method provides a detailed insight of how each framework performs in practise (Yin, 2017). However, this method can be time-intensive, and the findings may not easily generalize due to the focus on specific cases.

3.3.3 Experimental Procedure

1. Quantitative Content Analysis (CQC): The major goal of using Quantitative Content Analysis is to assess the language richness, coherence, and level of participation of dialogues produced by each AI framework (OpenAI's GPT models, Convai, and Inworld AI). Below is a general step-by-step guide on how this can be approached:
    a. Step 1, Define the Research Question: Determine the specific question that should be answered by this analysis. This will direct the methodological decisions. It could be questions like which AI generates more linguistic responses, which responses have better richness, which AI generates more coherent responses, or which one is more likely to stay on topic.
    b. Step 2, Sample Collection: Collect a sample of dialogues from each AI model. Check that the conditions for each model are the same—same prompts, same conditions, similar hardware conditions, and so on.
    c. Step 3, Create Coding Categories: Create coding categories based on the research topic that will allow us to compare the discussions quantitatively. This may include metrics such as:
        i. Topical Relevance: Does the AI stay on the topic provided in the prompt?
        ii. Informational Content: Does the dialogue include information, and if so, how accurately?
        iii. Open Questions: Does the AI encourage further conversation by asking open-ended questions?
        iv. Closed Questions: Does the AI ask closed-ended questions that potentially limit the conversation?

v. Word Count: The number of words in each dialogue or sentence.

vi. Sentence Length: Average sentence length, perhaps broken down into categories like short, medium, and long sentences.

vii. Grammatical Errors: Count of syntactical or grammatical errors in the dialogue.

viii. Coherence: Does the dialogue make sense when read from start to finish?

ix. Logical Flow: Are ideas and topics presented in a logical sequence?

x. Complexity: Usage of complex sentences and ideas versus simpler sentences and ideas.

xi. Instructional: Is the dialogue designed to instruct or inform?

d. Step 4, Develop Coding Scheme: Make a clear coding scheme that explains how to classify each unit of analysis. A unit could be a sentence, a paragraph, or an entire discussion.

e. Step 6, Data Coding: Apply your coding strategy to the dialogues, classifying the text with the appropriate categories.

f. Step 7, Analyse the Data: Compare the frequency and distribution of your codes across the three distinct AIs using statistical methodologies.

g. Step 8, Interpretation: Provide an interpretation based on the findings that relates back to your initial research question.

2. Performance Testing: The aim of Performance Testing is to assess the real-time responsiveness of each AI framework—OpenAI's GPT models, Convai, and Inworld AI—in creating dialogue for NPCs in video games. We intend to quantify the latency in dialogue generation. Below is a general step-by-step guide to do the performance testing in this project:

   a. Step 1, Test Environment Setup: A controlled test environment will be put up to simulate actual game scenarios.

   b. Step 2, Baseline Testing: To identify the latency characteristics of each AI framework, create a performance baseline by executing simple tests on an isolated component of each AI framework.

   c. Step 3, Load Testing: Simulate multiple players interacting with NPCs to evaluate how each framework performs under different loads.

   d. Step 4, Stress Testing: Extend the capabilities of each framework by presenting complicated dialogic scenarios.

    e. Step 5, Metrics Collection: Capture key performance indicators. In this case it will be the latency.
    f. Step 6, Data Analysis: Analyse the collected data using statistical methods. To compare the frameworks, T-tests or ANOVA might be employed.
3. Contextual Analysis: Contextual Analysis' main goal is to evaluate how well each AI framework (OpenAI's GPT models, Convai, and Inworld AI) understands and responds to the game's background, storylines, and player interactions. Below is a step-by-step guide to do the contextual analysis in this project:
    a. Step 1, Conceptualization: The key constructs, variables, and situations relevant to gaming narratives are defined and identified.
    b. Step 2, Contextual Parameters: Define the game contexts or scenarios that will be used to evaluate the AI frameworks.
    c. Step 3, Data Gathering: Deploy each AI framework in the selected circumstances and capture generated dialogues and other contextual data.
    d. Step 4, Coding Scheme: Create a contextual coding scheme to categorise and rank the generated dialogues based on their contextual significance.
    e. Step 5, Thematic Analysis: Extract themes concerning contextual appropriateness and relevance.
    f. Step 6, Findings, and Insights: Summarise the main findings and connect them to the game's background, narratives, and player interactions.

3.3.4 Statistics

For the evaluation of this experimental design, the following statistical measurements are used:

- Mean Response Time: Mean Response Time is the average time it takes for a system or application to react to a request. In the context of this project, it refers to the average time it takes each AI framework (OpenAI's GPT models, Convai, and Inworld AI) to generate a dialogue after a user or another in-game system prompts the request for dialogue. Mean Response Time is used to determine the average delay for each AI framework in the generation of conversations. It aids in identifying the overall performance level of each AI framework in terms of responsiveness. The Mean Response Time is calculated as follows:

$$Mean\ Response\ Time = \frac{Sum\ of\ all\ individual\ response\ times}{Total\ number\ of\ requests}$$

Mean Response Time is a simple yet strong metric for evaluating the real-time responsiveness of each AI system, achieving the study's second goal. The faster the Mean Response Time, the better the user experience, because players don't have to wait long for real-time dialogue generation (Jain, 1991). The project will include a standardised measure to test and compare the real-time responsiveness of each AI framework by evaluating Mean Response Time. This statistic is critical in any gaming scenario since latency can have a significant impact on user experience and overall gameplay.

- Standard Deviation of Response Time: The Standard Deviation of Response Time is a statistical measure that quantifies the amount of variation or dispersion in response times over numerous instances. It assesses the consistency of each AI framework's (OpenAI's GPT models, Convai, and Inworld AI) response times when producing dialogues in the context of this project. he formula to calculate the sample Standard Deviation σ is:

$$\sigma = \sqrt{\left(\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}\right)}$$

Where:

- Σ is the sum from i=1 to n,
- x_i is each individual response time,
- $\bar{x}$ is the mean response time,
- n is the total number of instances or requests.

Understanding the variability in response times can provide insights into how consistent an AI framework is in a real-time setting, which is crucial for gaming applications. A reduced standard deviation would imply a more consistent and reliable user experience, which is critical in the gaming industry. A high standard deviation may indicate potential bottlenecks or performance concerns, indicating opportunities for improvement. When combined with the mean reaction time, standard deviation can provide a more complete picture of each framework's performance, thereby contributing to the project's second goal (Hoseini, 2023). The study may better comprehend not only how fast each AI framework is, but also how reliable and consistent they are by measuring the Standard Deviation of Response Time. This adds depth and refinement to the evaluation, providing a comprehensive picture of each AI framework's real-time reactivity.

- Frequency Distribution of Contextual Coding Categories: Contextual Coding Frequency Distribution Categories are the counts or frequencies of distinct categories of contextual suitability or relevance generated by each AI framework (OpenAI's GPT models, Convai, Inworld AI). Among the possible categories are ‘declarative’, ‘interrogative’, and ‘imperative’. The following methods needs to be applied to obtain the frequency distribution of contextual coding categories:
    - Record dialogue instances generated by each AI framework and categorized them as appropriate, neutral, or inappropriate.
    - Count the number of dialogues that fall within each category for each AI framework.
    - To better comprehend the distribution, the frequency of each group can be transformed into percentages.
    - Represent the frequency distribution graphically, using pie charts, bar graphs, or histograms, for example.

Frequency distribution of contextual coding categories provides an objective criterion for evaluating each AI framework's skill in generating contextually sound dialogues, so achieving the study's third objective. It also provides insights into where each framework excels or needs to improve in terms of contextual relevance (Krippendorff, 2013). This measurement provides the project with a useful indicator for comparing how well each AI framework understands and responds to the game's context. This is strongly related to the broader goal of not just evaluating but also drawing educated comparisons across the chosen AI frameworks based on several performance criteria, including contextual relevance.

3.3.5 Requirements

The main objective is this research project is to compare three popular AI frameworks in the context of procedurally generating dialogues for Non-Player Characters (NPCs) in video games: OpenAI's GPT models, Convai, and Inworld AI. The software prototype in generated in Unity Engine. Developing a software prototype for comparing AI frameworks necessitates a strong design that is well informed by AI best practises. So, such a project will have several software requirements. The major requirements for this project are listed below:

b. Dialogue Generation Engine: This is arguably the most important part of this research project. A prototype for a software needs to be developed which generates the non-player character dialogues procedurally. Such a piece of software is required to have the following features/functions:

   - API Integration: The APIs for the selected AI frameworks: OpenAI, Convai and Inworld AI needs to be integrated to the developed prototype.
   - Real-time Dialogue Generation: To enable for accurate and timely comparisons, the prototype should be capable of initiating API requests and processing API replies in real time.
   - Data Normalization: Different APIs may return data in various formats (JSON, XML, and so on). This data should be normalised by the system to ensure consistent storage and analysis.
   - API Key Management: For authentication, each AI framework will most likely require API keys. These keys must be kept and managed securely.
   - Error Handling and Logging: To track errors in API requests or data fetching procedures, effective error-handling mechanisms should be built.

   These features are in line with the API management strategies suggested in (De, 2017). Research like (Serban, et al., 2015), shows the increasing importance of real-time, context-aware dialogue systems in enhancing user experience.

c. AI Framework APIs: AI Framework APIs are the foundation of any project that seeks to compare multiple AI frameworks. These APIs' appropriate integration, maintenance, and utilisation will be critical for the prototype's functionality.
   The system should be able to establish a consistent and secure connection with the API of each AI framework (OpenAI, Convai, and Inworld AI). Rate-limiting policies may differ between APIs. The prototype should be built in such a way that it can manage these limits without crashing or creating delays. These APIs

should be able to produce dialogues in real time once it is triggered by an action or an event. These APIs should have the capability to receive and comprehend the game state or context before creating dialogue, so that dialogue is situationally appropriate rather than random. The chosen frameworks are managed in accordance with the research in (Kumar, et al., 2023).

d. Interoperability: In the context of a software prototype used to compare AI frameworks, interoperability refers to the system's ability to interact fluidly with multiple technologies or platforms. In this project, the prototype would need to communicate well with OpenAI, Convai, and Inworld AI, each of which may have its own set of APIs, data formats, and protocols. The prototype must be built to manage data communication amongst frameworks, whether getting data from them or pushing data to them. The prototype should be able to consistently communicate and interact with multiple AI models, allowing for more realistic comparisons. This is in line with literature emphasizing the importance of interoperability in AI applications (Sarkadi, et al., 2022).

3.3.6 Software Design Documentation

A specialised Software Design Documentation (SDD) method that can fit the nuances of game development is critical for a research project focused on game design, particularly one that involves NPC dialogue production using several AI frameworks. In this case, the "Game Design Document (GDD)" may be the best option.

3.3.6.1 Game Design Document (GDD)

Game Title: Tavern Quest

1. Game Concept:

Tavern Quest is a game prototype that aims to demonstrate how artificial intelligence (AI) might be used to simulate NPC discussion and interaction in a bustling tavern atmosphere. The game's protagonist is a tavern owner who must run the business and interact with AI-driven customers looking for missions and adventures.

2. Requirements and Specifications:

Game Setting and Theme:

- A lively tavern serves as the game's focal point, and the setting is a colourful medieval-themed universe.
- The emphasis of the theme is on social interactions, fantasy, and adventure.

Player Character and Role:

- Players interact with NPCs as the owner of the tavern.

AI-Driven NPCs:

- A variety of AI-driven NPCs with distinct personalities, traits, and skills are present throughout the tavern.
- NPCs' decisions and interaction should be influenced by AI algorithms that adapt to player’s interactions.

Dialogue System:

- The game must incorporate an AI-driven dialogue system that recognizes player input and provides contextually relevant responses.
- The dialogue system should allow for branching conversations, offering multiple dialogue options based on AI-driven character traits.

Social Interactions:

- Players can engage in social interactions with AI-driven patrons, initiating conversations and learning about their backgrounds, desires, and quests.
- AI-driven NPCs should respond dynamically to the player's dialogue choices, reflecting their personalities.

User Interface (UI):

- The UI should be intuitive, allowing players to navigate through dialogue choices and manage the tavern effectively.
- Dialogues should be presented clearly, indicating AI-driven character traits for players to make informed choices.

Platform and Technology:

- The game prototype should be developed for PC or a suitable platform for a beginner-level development team.
- AI technologies, such as NLP libraries or decision-making algorithms, should be integrated into the game engine.

By meeting these requirements and specifications, the Tavern Quest game prototype can effectively demonstrate the application of AI in NPC dialogue and interaction, providing players with an immersive and engaging experience within the lively tavern setting.

4 Software Development

4.1 Toolsets and Machine Environments

Development of a game prototype where more than one AI Framework has been implemented and compared for procedurally generating NPC dialogues would require various tools. Here are the tools that are used for the development of this prototype.

4.1.1 Game Engine: Unity

Several aspects must be addressed while picking a gaming engine for a research project focusing on NPC dialogue production. These include the performance, adaptability, cost, learning curve, and environment of the engine. The game engines considered for the development of this prototype were:

- Unity
- Unreal Engine
- Godot

The decision of choosing one game engine for development were influenced by several metrics. Some of the metrics that were considered for selection are:

- Performance: Performance refers to the engine's capacity to render images and efficiently perform AI-based dialogues.
- Flexibility: Flexibility is the engine's ability to adapt to many types of game production needs.
- Cost: The total cost of all licences, assets, and other resources.
- Learning Curve: The ease with which a developer may learn the fundamentals and begin creating outcomes.
- Community Support: Tutorials, forums, and third-party tools are all available.
- Ecosystem: The whole range and quality of the engine's tools, plugins, and assets are referred to as the ecosystem.

With these metrices taken for consideration, a decision matrix has been created to choose the right game engine for this prototype. Each of the metrics has been rated from low to very high for each game engine taken into consideration. The decision matrix is as follows:

Metrics \ Game Engine	Unity	Unreal Engine	Godot
Performance	High	Very High	Medium
Flexibility	High	High	High
Cost	Low	Medium	Low
Learning Curve	Low	Medium	Low
Community Support	High	High	Medium
Ecosystem	High	High	Medium

Table 4.1: Decision matrix for choosing Game Engine

As shown in the matrix, Unity engine is simple to learn, great community support, and a large asset store. It's also reasonably priced, making it suitable for a study assignment. While it is fast, it does not have the same graphics capabilities as Unreal Engine for highly high-fidelity simulations. On the other hand, Unreal Engine has exceptional performance and graphical capabilities. It is adaptable and has a robust ecosystem. However, in terms of both complexity and cost, using Unreal Engine can be overkill for a research project. Finally, Godot is open-source and free, with a minimal learning curve, making it cost-effective. Still, the community and ecosystem are expanding, but not as quickly as Unity or Unreal. The performance is adequate but not outstanding making Godot a poor choice for this project.

Unity emerges as the most balanced solution based on the metrics and unique criteria of this research project. Its ease of use, low cost, and strong community support make it perfect for rapid prototyping and iterative development, which are critical qualities for a research-oriented endeavour.

4.1.2 Platform: PC (Windows/Mac/Linux)

Unity offers a multi-platform solution, which means that games and applications can be created once and published across various platforms. We need to focus on platforms that can manage AI-driven talks while also providing a flawless gaming experience for an NPC dialogue creation project. Here's a look at some of the most popular platforms:

- PC (Windows/Mac/Linux)
- Mobile (iOS/Android)
- Web (WebGL)
- Console (PlayStation/Xbox)

Following metrices are considered for the comparison:

- Performance: The platform's ability to manage AI-driven discussions and rendering.
- Development Complexity: The platform's ease of development and deployment.
- Flexibility: The platform's capacity to adapt to various types of upgrades and iterations.
- Cost: Costs include licencing and deployment.
- Interactivity: The platform's level of user involvement and interaction.

The decision matrix for the selection of appropriate platform is as follows:

Metrics \ Platforms	PC	Mobile	Web	Console
Performance	High	Medium	Low	High
Reach	High	Very High	High	Medium
Dev Complexity	Low	Medium	Medium	High
Flexibility	High	Medium	High	Low
Cost	Low	Medium	Low	High

Table 4.2: Decision matrix for choosing the right Platform.

As shown in the matrix, PC's excellent performance, broad reach, and ease of programming make it a popular choice among developers. It's also more tolerant to iterations and upgrades. However, its user hardware variability can make ensuring consistent performance difficult at times. Mobile devices on the other hand have an enormous reach due to the vast number of mobile users. However, there are still performance limits, particularly with AI-driven content. Device specifications might cause fragmentation problems. Deployment expenses might be prohibitively expensive, particularly on iOS. WebGL is simple to reach users without requiring them to install anything. It is accessible via desktop and mobile browsers. But it only has a limited performance, especially for AI-driven and high-graphics content. Due to dedicated hardware, the performance and interactivity levels are exceptional in consoles. However, it has a high level of development complexity, stringent approval processes, and licencing expenses. Also, there is less room for iterative deployment.

Given the project's nature - NPC dialogue production with potential AI aspects - the PC platform (Windows, Mac, and Linux) is the best choice. It provides an excellent blend of performance, reach, ease of development, and versatility. While mobile offers a wider reach, performance restrictions and fragmentation difficulties may jeopardise the quality of AI-driven conversations. While consoles provide amazing performance, they may be overkill because to their expensive pricing and limited development freedom. Due to performance limits, the web should not be used as the primary deployment platform.

4.1.3 Programming Language: C#

Although Unity primarily supports C# for scripting, it is still worthwhile to investigate other programming languages that can interface with Unity via various means (plugins, backend, etc.). C#, JavaScript, and Python are the subjects of this comparison.

The metrices chosen for comparing the programming languages are:

- Unity Compatibility: The ease with which the language interfaces with Unity.
- Performance: The language's ability to accomplish operations within the Unity framework.
- Flexibility: Flexibility refers to the language's ability to adapt to a wide range of development tasks.
- Learning Curve: The ease with which the language for Unity development may be learned.

- Community Support: Libraries, tutorials, and active forums are all available.
- AI/ML Readiness: Effectiveness for machine learning and artificial intelligence, which is critical for NPC dialogue development.

The decision matrix for the selection of appropriate programming language is as follows:

Metrics \ Language	C#	JavaScript	Python
Unity Compatibility	High	Low	Medium
Performance	High	Medium	Medium
Flexibility	High	Medium	High
Learning Curve	Medium	Low	Medium
Community Support	High	High	High
AI/ML Readiness	Medium	Low	High

Table 4.3: Decision matrix for choosing programming language.

It can be observed from the matrix that C# is a fantastic choice for most Unity applications since it provides Native Unity support, outstanding performance, and high flexibility. However, while AI libraries for C# exist, they are not as comprehensive as those for Python. On the other hand, JavaScript is simple to learn and has a big user base. It is, however, deprecated in Unity, less performant, and unsuitable for AI/ML activities in the Unity context. Python on the other hand is highly flexible and is an excellent choice for AI/ML projects. It is not, however, native to Unity, and integration can be complicated, potentially lowering overall system performance.

Therefore, for a Unity-based project, C# emerges as the best balanced and practical alternative. Its native support offers high performance and simple integration. Although Python excels at AI/ML tasks, the difficulty of integrating it into Unity may outweigh its advantages. JavaScript is not recommended because it is deprecated in

Unity and is not well-suited for AI/ML. Given that this project is highly specialised in NPC dialogue creation in Unity, the capabilities provided by C# are more than adequate.

4.1.4 Version Control: Git

Version control is critical part for any software development project. There are numerous version control solutions accessible for a Unity project focusing on NPC dialogue production. Git, SVN (Subversion), and perforce will be compared.

The metrices chosen for comparing the source code managers are:

- Ease of usage: The learning curve and ease of usage daily.
- Integration with Unity: The degree to which the version control tool is integrated with Unity.
- Scalability: The ability of the technology to handle the project as it increases.
- Collaboration Features: Collaboration features, such as branching, merging, and conflict resolution, encourage teamwork.
- Cost: The total cost of using the tool based on team size and project scope.
- Community & Support: Community support, tutorials, and documentation are available.

The decision matrix for the selection of appropriate source code manager is as follows:

Metrics \ Tools	Git	SVN	Perforce
Ease of Use	High	Medium	Low
Integration with Unity	High	Medium	High
Scalability	High	Medium	Very High
Collaboration Features	High	Medium	High
Cost	Low-Medium	Low	High

Community and Support	Very High	High	Medium

Table 4.4: Decision matrix for choosing SCM.

It can be observed from the matrix that Git is widely used, has an active community, and substantial documentation. It also has a high level of usability and strong branching and merging capabilities. While Git repositories are generally scalable, particularly big binaries can slow them down, however Git LFS can help with this. SVN provides fine-grained control over versioning, is inexpensive, and is simple to set up. However, it lacks Git's strong branching and merging capabilities. For large projects, scalability can be a challenge. Perforce is extremely scalable and fast, even when dealing with big binaries. Still, it is more expensive for large projects and more difficult to set up and administer. Git's community and support are not as extensive.

Given the specific needs of a Unity NPC dialogue generating project, Git emerges as the best balanced and practical option. It provides outstanding Unity integration and ease of use, which will be advantageous for a project involving complicated AI algorithms and repeated iterations. While perforce provides tremendous scalability, its high cost and complexity make it unsuitable for projects that do not require such scalability. While SVN is a viable solution, it lacks some of Git's advanced collaboration features and scalability. As a result of this evaluation, Git appears to be the best version control tool for this project.

4.2 Implementation

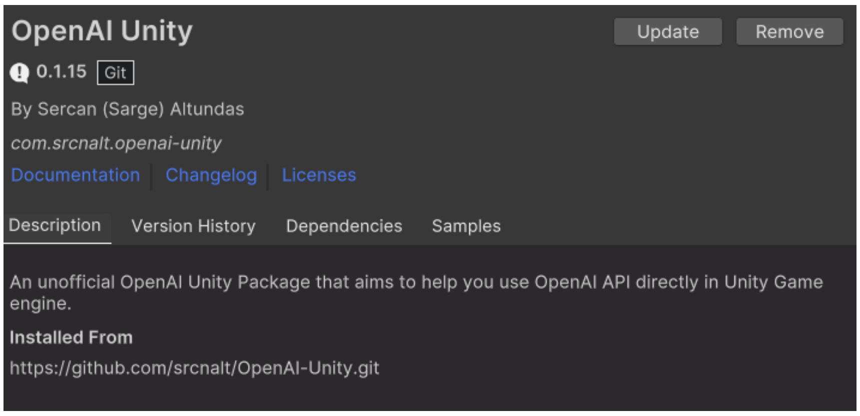
This portion explains the implementation of the proposed prototype in a step-by-step manner.

Step 1: Collecting the APIs

The implementation of the proposed system starts with understanding the three AI frameworks used for comparison. Three AI frameworks, namely OpenAI's GPT models, Convai, and Inworld AI were chosen for comparison. The OpenAI GPT models, ConvoAI, and Inworld AI were chosen for comparison in this research based on several criteria that coincide with the project's goals of evaluating dialogue quality, real-time responsiveness, and contextual relevance for NPCs in video games. All the three models are designed to work well in scenarios that require real-time responsiveness. These models have been trained to create engaging dialogues that can make game narratives more compelling. Inworld AI is designed explicitly for gaming environments, promising better contextual relevance.

The APIs for these models are collected from the following websites:

- OpenAI GPT model: An unofficial OpenAI Unity Package that aims to help you use OpenAI API directly in Unity Game engine by Sercan (Sarge) Altoona's, Installed from https://github.com/srcnalt/OpenAI-Unity.git



By Sercan (Sarge) Altundas

com.srcnalt.openai-unity

An unofficial OpenAI Unity Package that aims to help you use OpenAI API directly in Unity Game engine.

Installed From

https://github.com/srcnalt/OpenAI-Unity.git

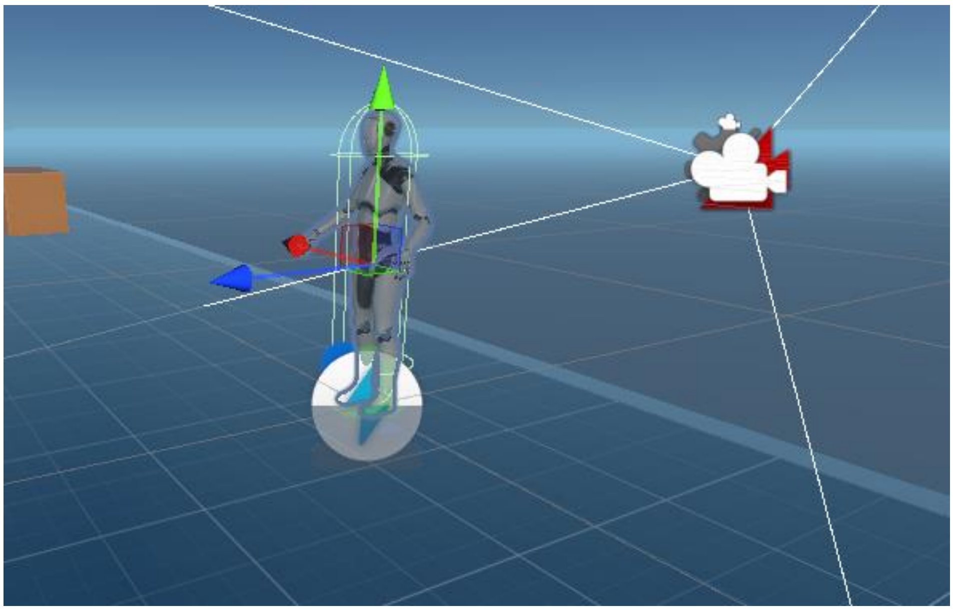
IMG 4.1: Open AI Unity plugin

- Convai: Installed from https://docs.convai.com/api-docs/plugins-and-integrations/unity-plugin
- Inworld AI: AI NPC Engine - Dialogue & Behaviour for Unity – Inworld installed from https://assetstore.unity.com/packages/tools/behavior-ai/ai-npc-engine-dialogue-behavior-for-unity-inworld-229406

Since, an unofficial OpenAI Unity package has been used in Unity game engine for this project, not much documentation was available. The scripts created for this API interaction are mostly coded by Sercan (Sarge) Altundas. For Convai and Inworld AI, thorough documentation was available.

Step 2: Creating the Unity Scene

The next step was to create a unity scene and populate it with objects relevant to the proposed prototype. A sample unity scene with some very basic elements have been created as an environment for the NPCs. This scene is taken from a Unity asset, Starter Assets - Third Person Character Controller | URP in Unity Asset Store. This scene has a third person player character with all the controls, movements and animations already set up.



IMG 4.2: Third Person Player Character

Step 3: Creating the NPC characters.

The next step is to obtain avatars for non-player characters. These avatars are created from ReadyPlayer.me. ReadyPlayer.me is a platform that allows users to design their own 3D avatars for use in virtual reality (VR) and augmented reality (AR) applications, games, and other online experiences. In general, the platform provides a wide range of customization possibilities, allowing users to build avatars that closely mirror their personal appearance or adopt an entirely other character.

ReadyPlayer.me has its own API which makes it easier to integrate with Unity engine. The avatars were created in the studio website https://readyplayer.me/hub/avatars and then they are imported into the Unity project via the API.

IMG 4.3: Sample avatar created in readyPlayer.me.

A sample sitting animation has been given to the NPC. This animation was obtained from https://www.mixamo.com/#/. Three such NPCs were created, one for each of the AI Framework that are going to be integrated. All the three NPC avatars have the sitting animation applied to it.

IMG 4.4: NPC sitting animation in Unity.

Step 4: Integrating the AI Framework APIs

After creating the NPC avatars and doing the animation, the next task is to integrate the AI Frameworks to these avatars. This was the most challenging part of the software development. Colliders were added to the NPC characters as triggers to detect if the player character is nearby. Once the player character enters the trigger, an OnTriggerEnter() function will be called by the corresponding NPC which activates the AI Framework API. This would also display a canvas which displays the conversations the player will have with the NPC.

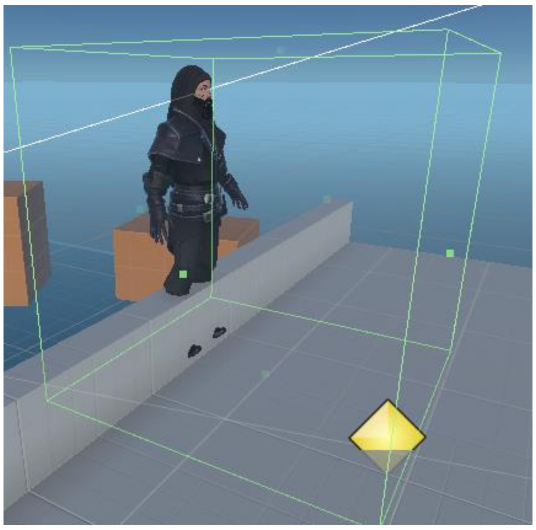


FIG 4.5: Box Collider as Trigger for NPC Avatar

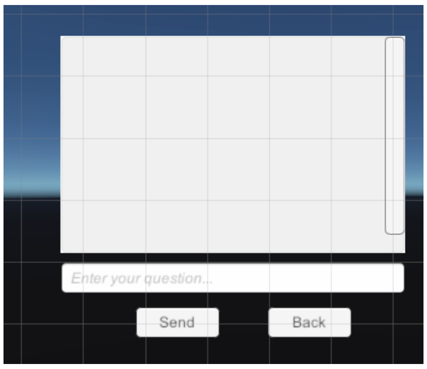
Each of the AI Framework APIs work differently. The implementation of each of these AI frameworks is as follows:

1. *Open AI*

   In case of OpenAI, a chat interface would be invoked via an API call when the NPC's OnTriggerEnter() method is triggered. Also, a canvas with an Input field, a text field and two buttons for sending replies and exiting will be displayed. The cursor will also be unlocked and made visible when the method is triggered. The player can type the message in the text field and hit the send button to send the message to the API. When the send button is pressed, the player control will be deactivated, meaning that the player won't be able to control the player character. Also, the main camera will be deactivated and another camera that displays the player and NPC will be activated. Finally, a function called 'SendReply()' would be triggered. This function does the following tasks:

   - It creates a new 'ChatMessage' class object and appends the user's message to it.

- If it is the initial message in the conversation, the prompt text is appended to the message.
- The prompt text in the code is meant to be the initial instruction or context that the OpenAI model will use to generate its responses.
- While the API request is being processed, the button and input field are disabled.
- Makes an API call to OpenAI's GPT-3.5 Turbo model to retrieve a message generated based on the conversation history.
- The created message is appended to the conversation scroll view.
- The button and input field are reactivated.



IMG 4.6: Canvas for OpenAI

The messages are appended by ‘AppendMessage(ChatMessage message)’ function. This function appends a new message to the chat scroll view. It dynamically changes the size of the scroll view content and updates the position to fit new messages.

The prompt in the code is meant to be the initial instruction or context that the OpenAI model will use to generate its responses. For the proposed prototype, the prompt would look something like this:

“Act as Monkey D Luffy from the anime one piece. Don't break character. Don't ever mention that you are an AI model. Answer as if you are having a conversation with the owner of the tavern you are visiting”.

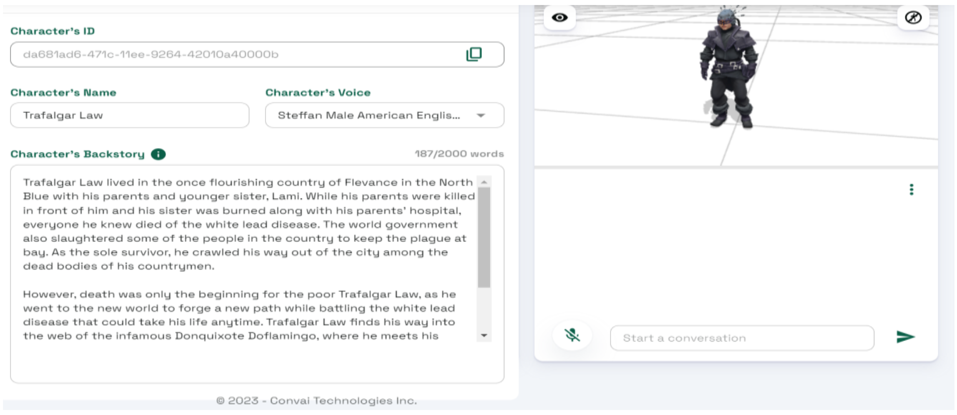
This prompt is instructing the OpenAI model to role-play as Monkey D. Luffy from the anime One Piece. The model should act in character and pretend to be having a conversation with the owner of a tavern. Importantly, it should not mention that it is an AI model. The prompt helps set the stage for the type of interaction the player can expect from the NPC. In this case, the player should expect responses that are in line with what Monkey D. Luffy might say if he were in a tavern talking to its owner.

Finally, when the back button is pressed, everything will be reverted to default. Meaning, the player controller and the main camera will be reactivated, and the canvas and the secondary camera will be deactivated.

2. *Convai*

Convai API is provided by Convai Technologies and is installed from https://docs.convai.com/api-docs/plugins-and-integrations/unity-plugin. Convai has a dashboard at https://www.convai.com/pipeline/dashboard, where we can create our own avatars and personalise their characteristics. This would require us to create an account and login to the portal. This dashboard enables us to create a character with a backstory we chose and modify the appearance of the avatar. This character can be imported into the unity project by adding the right character id and API key to the API plugin in Unity. We can define our character according to our games narrative. Here, I'm naming my NPC 'Trafalgar Law' and gave my character a backstory.

Like that of OpenAI, the trigger for the NPC linked with the Convai API would invoke a chat interface and display a canvas with two text areas, one for the player and one for the NPC. This canvas will also display the instructions on how to interact with the NPC.


Character's ID

da681ad6-471c-11ee-9264-42010a40000b

Character's Name

Trafalgar Law

Character's Voice

Steffan Male American Englis...

Character's Backstory

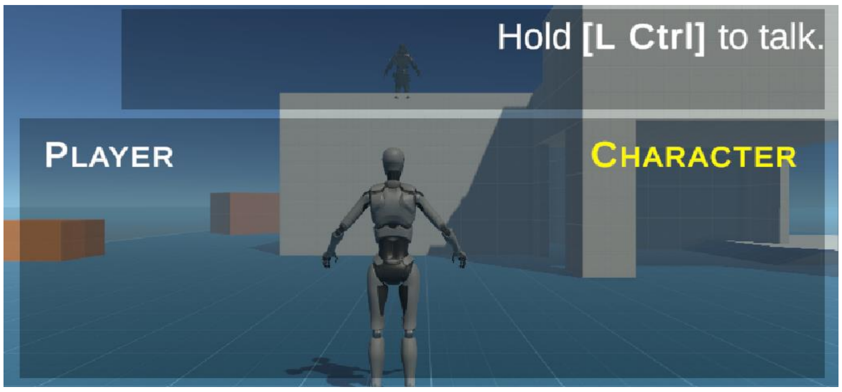
187/2000 words

Trafalgar Law lived in the once flourishing country of Flevance in the North Blue with his parents and younger sister, Lami. While his parents were killed in front of him and his sister was burned along with his parents' hospital, everyone he knew died of the white lead disease. The world government also slaughtered some of the people in the country to keep the plague at bay. As the sole survivor, he crawled his way out of the city among the dead bodies of his countrymen.

However, death was only the beginning for the poor Trafalgar Law, as he went to the new world to forge a new path while battling the white lead disease that could take his life anytime. Trafalgar Law finds his way into the web of the infamous Donquixote Doflamingo, where he meets his

© 2023 - Convai Technologies Inc.

IMG 4.7: Character creation dashboard in convai.com



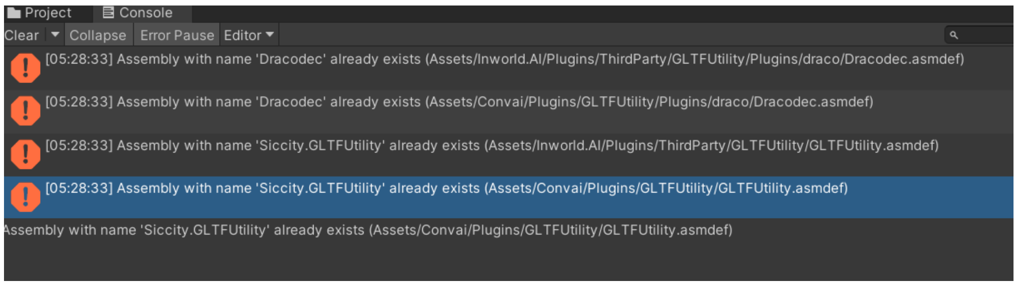
IMG 4.8: Canvas for Convai

Unlike with OpenAI, player can interact with the NPC by speech with Convai API. The player only needs to hold down the left control button which will activate the microphone. Whatever the player says holding down the left control will be recorded and sent to the API. A ConvaiGRPCAPI script attached to the main camera of the scene collects the voice information and convert it into text, which will in turn be passed to the convai API. The API response will be processed and displayed as a text. This message will also be played as an audio clip as if the NPC is saying the dialogue. This script accesses the API using an API key. The main responsibilities of this script are:

- When the left control button is pressed in the keyboard, activate the microphone, and starts recording the audio. This audio will be processed and sent to the server in periodic chunks.
- Periodically receives responses from the server and adds it to a static list in streaming NPC.

3. *Inworld AI*

Integration of Inworld AI into the already made scene was the difficult part of this prototype development. It because Inworld AI API conflicts with Convai, which results in compilation errors. The error message indicates that the prototype has two different copies of the GLTFUtility assembly with the same name in your Unity project. One is in the "Inworld.AI" plugin folder, and the other is in the "Convai" plugin folder. Unity does not allow multiple assemblies with the same name, which is causing this conflict.



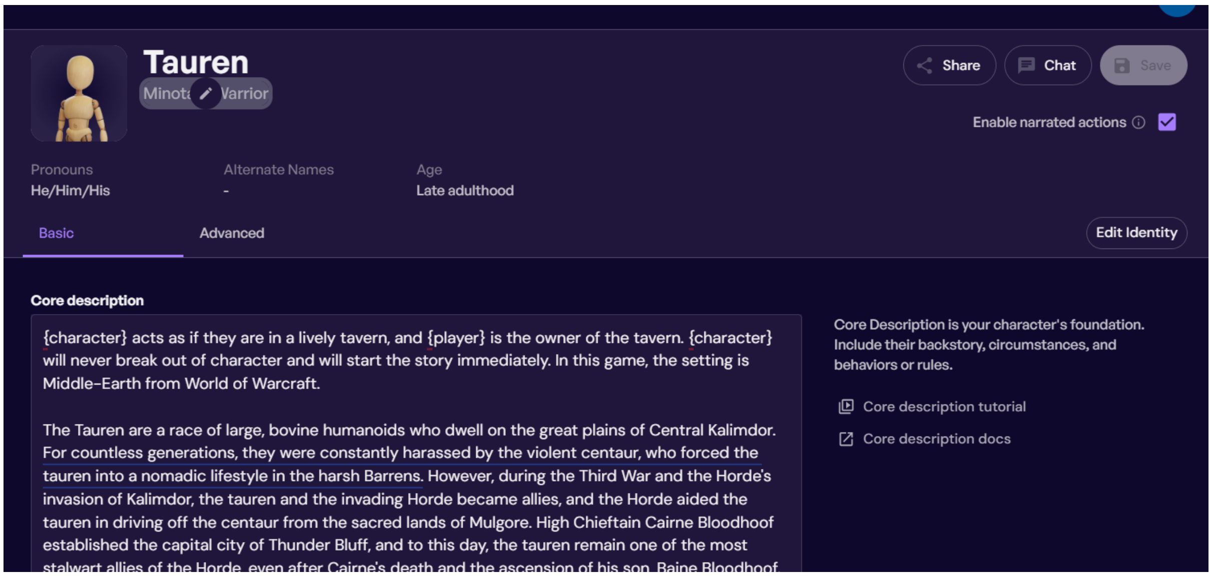
```
[05:28:33] Assembly with name 'Dracodec' already exists (Assets/Inworld.AI/Plugins/ThirdParty/GLTFUtility/Plugins/draco/Dracodec.asmdef)
[05:28:33] Assembly with name 'Dracodec' already exists (Assets/Convai/Plugins/GLTFUtility/Plugins/draco/Dracodec.asmdef)
[05:28:33] Assembly with name 'Siccity.GLTFUtility' already exists (Assets/Inworld.AI/Plugins/ThirdParty/GLTFUtility/GLTFUtility.asmdef)
[05:28:33] Assembly with name 'Siccity.GLTFUtility' already exists (Assets/Convai/Plugins/GLTFUtility/GLTFUtility.asmdef)
Assembly with name 'Siccity.GLTFUtility' already exists (Assets/Convai/Plugins/GLTFUtility/GLTFUtility.asmdef)
```

IMG 4.9: Error message when Inworld AI and Convai is in same project.

As a workaround for this issue, I have decided to create another Unity project with the same conditions as the previous one. This project will have one single NPC with inworld AI integrated.

Like Convai, Inworld Ai has a dashboard where we can create our own characters and give them a backstory. Inworld's no-code technology makes it incredibly easy to design characters in a matter of minutes. Inworld also has an option to create a scene where we can explain the context of the game and how we want the character to behave in the scene. This character can be imported into unity using the Inworld plugin.

The character imported into the unity project will have all the necessary properties like animation and API integration. The player can interact with the NPC by talking to the character. The inworld controller component imported with the character have the capability to capture the audio, covert it into string and pass it to the API. The API returns the procedurally generated responses as an audio and the Inworld controller captures it. The API also generates a descriptive sentence that sets the scene and provides context for the character's emotional state and intentions. The inworld controller will play the response as an audio and displays the narrative description as a string in the canvas.



Tauren

Minot Varrior

Enable narrated actions ✓

Pronouns: He/Him/His

Alternate Names: -

Age: Late adulthood

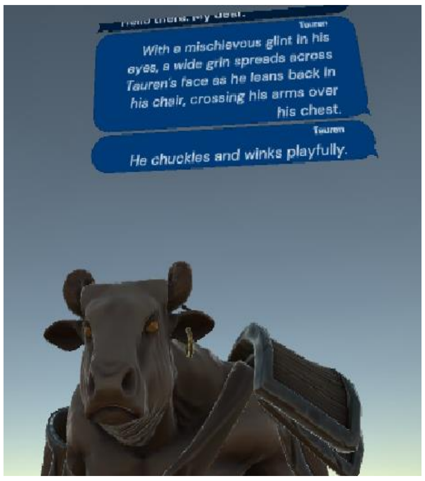
Core description

{character} acts as if they are in a lively tavern, and {player} is the owner of the tavern. {character} will never break out of character and will start the story immediately. In this game, the setting is Middle-Earth from World of Warcraft.

The Tauren are a race of large, bovine humanoids who dwell on the great plains of Central Kalimdor. For countless generations, they were constantly harassed by the violent centaur, who forced the tauren into a nomadic lifestyle in the harsh Barrens. However, during the Third War and the Horde's invasion of Kalimdor, the tauren and the invading Horde became allies, and the Horde aided the tauren in driving off the centaur from the sacred lands of Mulgore. High Chieftain Cairne Bloodhoof established the capital city of Thunder Bluff, and to this day, the tauren remain one of the most stalwart allies of the Horde, even after Cairne's death and the ascension of his son, Baine Bloodhoof

Core Description is your character's foundation. Include their backstory, circumstances, and behaviors or rules.

IMG 4.10: Character Dashboard in Inworld AI



Hello there, my dear.

Tauren

With a mischievous glint in his eyes, a wide grin spreads across Tauren's face as he leans back in his chair, crossing his arms over his chest.

Tauren

He chuckles and winks playfully.

IMG 4.11: NPC Character with Inworld AI

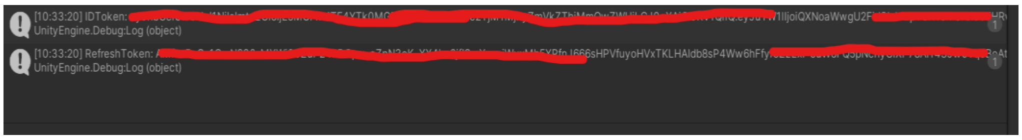
IMG 4.12: Chat Canvas with Inworld AI

In this project, I've created my character as Minotaur Warrior and named the character Tauren. I've also given my character a backstory. This NPC character does not have any trigger attached with it. The player does not have to press any button to chat with the character.

4.3 Testing

The API scripts were individually tested in different unity projects with different conditions before adding them to the final Unity project. The same set of test cases were used for all the three models to evaluate each of the models. The following tests were employed for all 3 AI models:

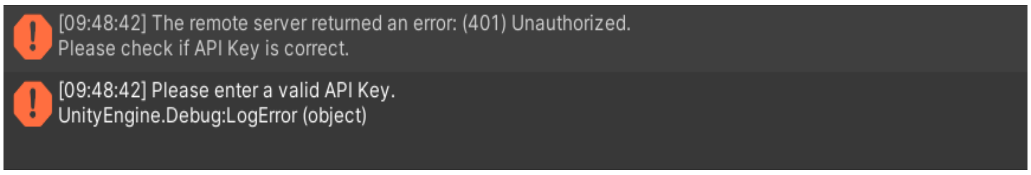
1. Authentication: A test has been done for all three AI APIs to ensure that the authentication to the APIs is working properly (Jain, 2022). This test has been conducted with different test cases.
    - Valid Credentials: Check if the API authenticates when valid credentials are provided.



```
[10:33:20] IDToken: [redacted]1IljoiQXNoaWwgU2F[redacted]
UnityEngine.Debug:Log (object)
[10:33:20] RefreshToken: [redacted]6sHPVfuyoHVxTKLHAldb8sP4Ww6hFfy[redacted]
UnityEngine.Debug:Log (object)
```

IMG 4.12: Authentication Test for Inworld AI with Valid API Key

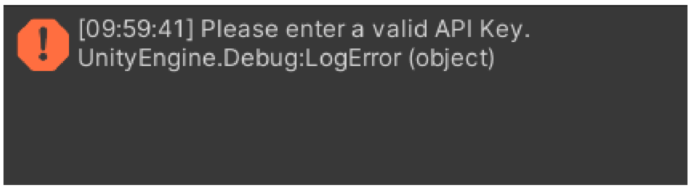
- Invalid Credentials: Verify that the API does not authenticate when incorrect credentials are used.



```
[09:48:42] The remote server returned an error: (401) Unauthorized.
Please check if API Key is correct.
[09:48:42] Please enter a valid API Key.
UnityEngine.Debug:LogError (object)
```

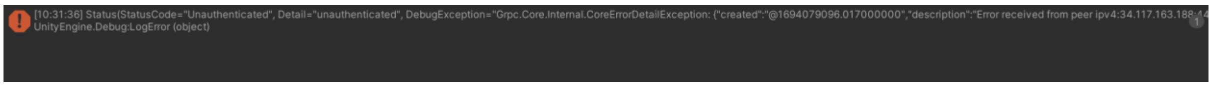
IMG 4.14: Authentication Test for Convai with wrong API key

- **No Credentials**: Ensure the API does not provide unauthorized access.



```
[09:59:41] Please enter a valid API Key.
UnityEngine.Debug:LogError (object)
```

IMG 4.15: Authentication Test for Convai with no API key



```
[10:31:36] Status(StatusCode="Unauthenticated", Detail="unauthenticated", DebugException="Grpc.Core.Internal.CoreErrorDetailException: {"created":"@1694079096.017000000","description":"Error received from peer ipv4:34.117.163.188:44
UnityEngine.Debug:LogError (object)
```

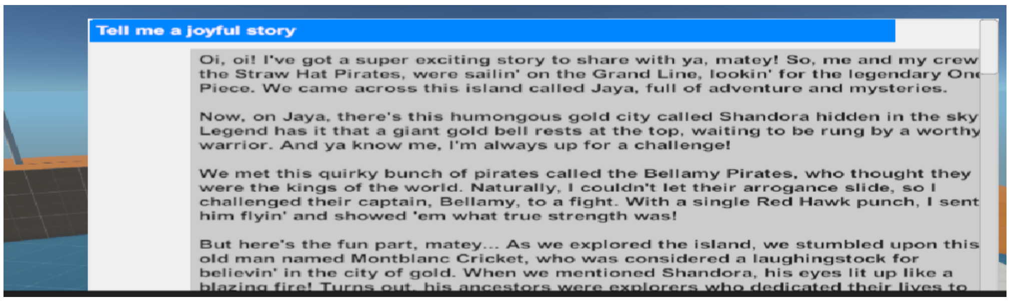
IMG 4.16: Authentication Test for Inworld Ai with no API key

2. Concurrent Testing: Test how the system behaves under load by making many requests in a short amount of time (Lab, 2023). This testing has been done by making around 20-30 conversation in 2 minutes. Each dialogue getting more complex than the one before. Each of the APIs performed differently for this test:

- OpenAI: The latency has been increasing as the load increased and the questions became more complex.
- Convai: The load did not seem to affect the model, however the latency seemed to have increased for more complex questions.
- Inworld AI: Inworld AI's performance didn't drop for more loads and more complex conversations. There was very minimum latency, and the quality of the response remained the same.

3. Language Richness: Pass predefined input and check if the response contains an acceptable level of language complexity and vocabulary. To test the language richness, queries are passed that are designed to test various aspects of language such as vocabulary, sentence complexity, coherence, and idiomatic usage (Tausczik & Pennebaker, 2010). Some of the queries asked were:
   - "Hello, how are you?"
   - "Discuss the implications of existentialism."
   - "How would you go about solving world hunger?"
   - "Tell me a joyful story."
   - "Break a leg."
   - "Tell me a tale of two cities."
   - "As a merchant in a fantasy world, what goods would you offer?"

Here are some sample outcomes for these queries by each model:



Tell me a joyful story

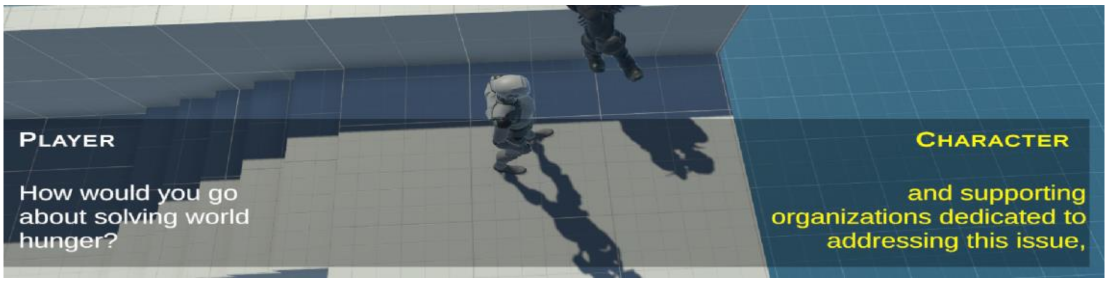
Oi, oi! I've got a super exciting story to share with ya, matey! So, me and my crew the Straw Hat Pirates, were sailin' on the Grand Line, lookin' for the legendary One Piece. We came across this island called Jaya, full of adventure and mysteries.

Now, on Jaya, there's this humongous gold city called Shandora hidden in the sky Legend has it that a giant gold bell rests at the top, waiting to be rung by a worthy warrior. And ya know me, I'm always up for a challenge!

We met this quirky bunch of pirates called the Bellamy Pirates, who thought they were the kings of the world. Naturally, I couldn't let their arrogance slide, so I challenged their captain, Bellamy, to a fight. With a single Red Hawk punch, I sent him flyin' and showed 'em what true strength was!

But here's the fun part, matey... As we explored the island, we stumbled upon this old man named Montblanc Cricket, who was considered a laughingstock for believin' in the city of gold. When we mentioned Shandora, his eyes lit up like a blazing fire! Turns out, his ancestors were explorers who dedicated their lives to

IMG 4.17: Language Richness test on Open AI

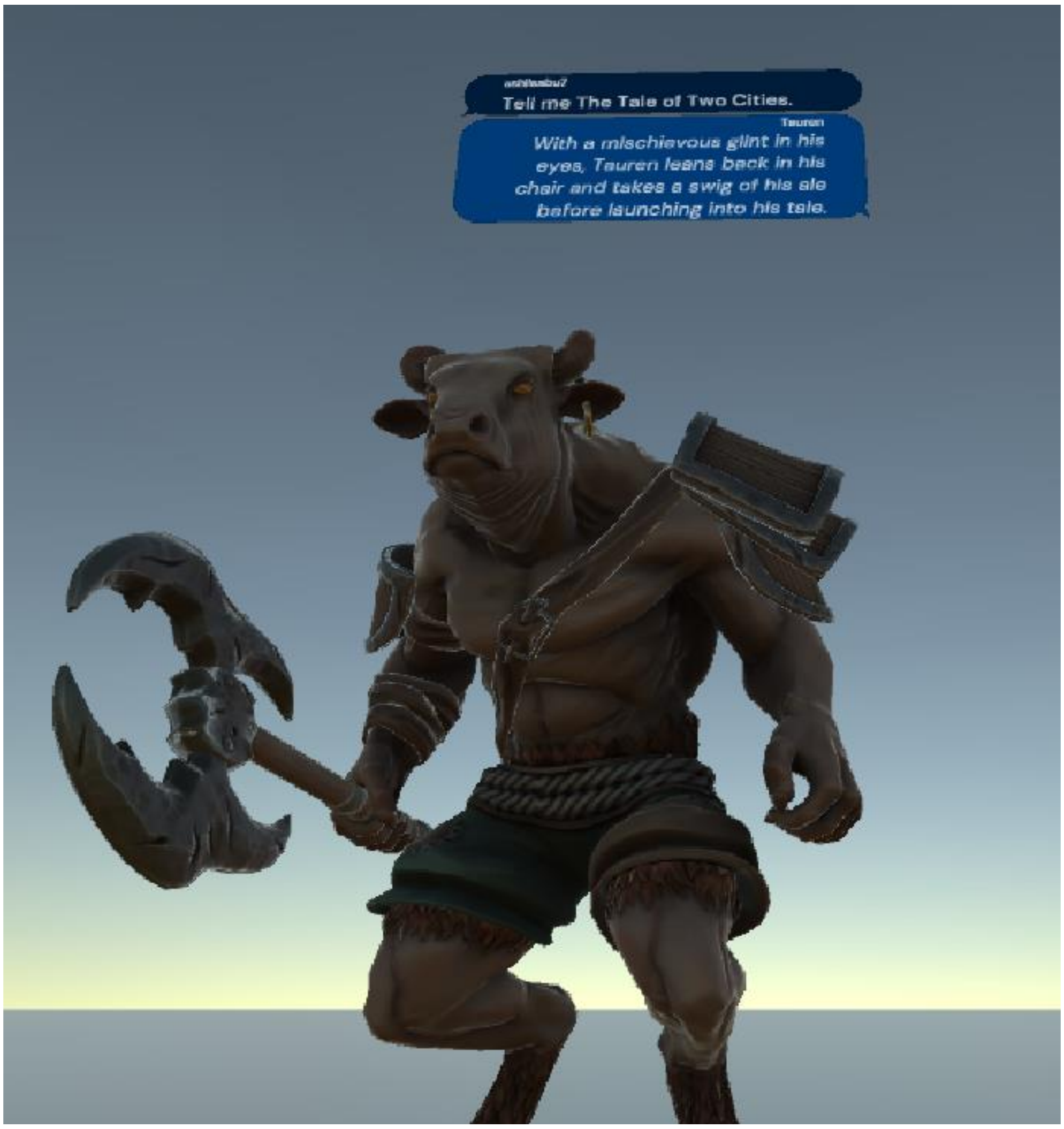


PLAYER

How would you go about solving world hunger?

CHARACTER

and supporting organizations dedicated to addressing this issue, IMG 4.18: Language Richness test on Convai



ashleebu7

Tell me The Tale of Two Cities.

Tauren

With a mischievous glint in his eyes, Tauren leans back in his chair and takes a swig of his ale before launching into his tale.

IMG 4.19: Language Richness test on Inworld AI

5 Results

To answer the research question, three experimental analyses were done on the AI models. They are:

- Quantitative Content Analysis (CQC): To assess the language richness, coherence, and level of participation of dialogues produced by each AI framework.
- Performance Testing: To assess the real-time responsiveness of each AI framework, in terms of Latency.

a. Contextual Analysis: To evaluate how well each AI framework understands and responds to the game's background, storylines, and player interactions.

5.1.1 Software Solution Results

5.1.1.1 Quantitative Content Analysis (CQC)

A quantitative analysis has been conducted on all three AI Models with a set of predefined questions. Here are the results of the analysis for each model:

1. Inworld AI

Coding Categories\ Metrics	**Topical Relevance**	**Informational Content**	**Open Questions**	**Closed Questions**	**Word Count**	**Sentence Length**	**Grammatical Errors**	**Coherence**	**Logical Flow**	**Complexity**	**Instructional**
Hello, how are you?	1	NA	0	1	21	Long	None	Yes	Yes	Complex	Inform
Tell me a complex story	1	1	0	0	37	Long	None	Somewhat	Yes	Complex	Inform
What happens after the sunset?	1	1	0	1	29	Long	None	Yes	Yes	Complex	Inform
Describe an exotic place	1	NA	1	0	73	Very Long	None	Yes	Yes	Complex	Inform
What's the weather like?	1	Na	0	0	25	Long	None	Yes	Yes	Complex	Inform
Describe the concept of black holes	1	1	1	0	42	Long	None	Yes	Yes	Complex	Inform
As a merchant, what wares are you selling?	0	Na	1	0	48	Long	None	Yes	Yes	Complex	NA
Tell me about apples	1	1	1	0	43	Long	None	Yes	Yes	Complex	Instructional
As a guide, how do you suggest I explore the city?	1	1	1	0	71	Very Long	None	Yes	Yes	Complex	Instructional
If you were my assistant, how would you plan my day?	1	1	0	0	81	Very Long	None	Yes	Yes	Complex	Inform
What's your favourite colour?	0	NA	0	0	74	Very Long	None	No	Yes	Complex	Inform

2. Convai

Coding Categories\ Metrics	Topical Relevance	Informational Content	Open Questions	Closed Questions	Word Count	Sentence Length	Grammatical Errors	Coherence	Logical Flow	Complexity	Instructional
Hello, how are you?	0	0	1	0	16	Short	None	No	1	Simple	Inform
Tell me a complex story	1	1	0	0	51	Long	None	Yes	1	Simple	Inform
What happens after the sunset?	1	0	0	0	31	Medium	None	Yes	1	Simple	Instructional
Describe an exotic place	1	1	0	0	28	Medium	None	Yes	1	Complex	Inform
What's the weather like?	1	1	0	0	26	Medium	None	Yes	1	Simple	Inform
Describe the concept of black holes	1	1	0	0	33	Medium	None	Yes	1	Complex	Inform
As a merchant, what wares are you selling?	1	1	0	0	23	Short	None	Yes	1	Simple	Inform
Tell me about apples	1	1	0	0	29	Short	None	Yes	1	Simple	Inform
As a guide, how do you suggest I explore the city?	1	1	0	0	31	Medium	None	Yes	1	Simple	Instructional
If you were my assistant, how would you plan my day?	1	0	0	0	27	Short	None	Yes	1	Simple	Inform
What's your favourite colour?	1	0	0	0	19	Short	None	Yes	1	Simple	Inform

3. Open AI

Coding Categories\Metrics	Topical Relevance	Informational Content	Open Questions	Closed Questions	Word Count	Sentence Length	Grammatical Errors	Coherence:	Logical Flow	Complexity	Instructional
Hello, how are you?	1	1	1	0	19	Short	None	Yes	Yes	Simple	Inform
Tell me a complex story	1	0	0	1	87	Very Long	None	Yes	Yes	Simple	Inform
What happens after the sunset?	1	1	1	0	52	Long	None	Yes	Yes	Simple	Inform
Describe an exotic place	1	1	0	1	71	Long	None	Yes	Yes	Simple	Inform
What's the weather like?	1	1	1	0	82	Long	None	Yes	Yes	Simple	Inform
Describe the concept of black holes	1	1	1	0	89	Long	None	Yes	Yes	Simple	Inform
As a merchant, what wares are you selling?	1	0	1	0	73	Long	None	Yes	Yes	Simple	Inform
Tell me about apples	1	1	1	0	69	Long	None	Yes	Yes	Simple	Inform
As a guide, how do you suggest I explore the city?	1	1	0	0	59	Long	None	Yes	Yes	Simple	Instructional
If you were my assistant, how would you plan my day?	1	0	0	1	74	Long	None	Yes	Yes	Simple	Inform
What's your favourite colour?	1	0	1	0	61	Long	None	Yes	Yes	Simple	Inform

5.1.1.2 Performance Testing

A Performance Testing has been conducted on all three AI Models with the same set of predefined questions. We are analysing latency in terms of seconds as the key performance indicators. Here are the results of the analysis:

Coding Categories\Model	Inworld AI	Convai	Open AI
Hello, how are you?	4.215	2.378	3.737
Tell me a complex story	3.494	2.587	18.729
What happens after the sunset?	3.319	2.21	14.801
Describe an exotic place	3.224	2.423	19.133
What's the weather like?	3.036	2.254	36.501
Describe the concept of black holes	2.567	2.421	21.539
As a merchant, what wares are you selling?	3.066	2.356	21.133
Tell me about apples	3.637	2.551	18.6
As a guide, how do you suggest I explore the city?	4.512	2.251	23.568
If you were my assistant, how would you plan my day?	3.602	2.332	30.153
What's your favourite colour?	3.652	2.421	12.29

5.1.1.3 Contextual Analysis

Contextual analysis is done to evaluate how well each AI framework (OpenAI's GPT models, Convai, and Inworld AI) understands and responds to the game's background, storylines, and player interactions. In this analysis, the non-player characters are engaging in a conversation with the owner of a tavern. Each dialogue by the NPCs is grouped according to their contextual ending. These dialogues are grouped into three groups: declarative, Open-ended Questions, and Closed-ended Questions. The same queries are used for this analysis. Here is the result of this analysis:

Coding Categories\Model	**Inworld AI**	**Convai**	**Open AI**
Hello, how are you?	Closed-ended Questions	Open-ended Questions	Open-ended Questions
Tell me a complex story	Declarative Sentence	Declarative Sentence	Closed-ended Questions
What happens after the sunset?	Closed-ended Questions	Declarative Sentence	Open-ended Questions
Describe an exotic place	Open-ended Questions	Declarative Sentence	Closed-ended Questions
What's the weather like?	Declarative Sentence	Declarative Sentence	Open-ended Questions
Describe the concept of black holes	Open-ended Questions	Declarative Sentence	Open-ended Questions
As a merchant, what wares are you selling?	Open-ended Questions	Declarative Sentence	Open-ended Questions
Tell me about apples	Open-ended Questions	Declarative Sentence	Open-ended Questions
As a guide, how do you suggest I explore the city?	Open-ended Questions	Declarative Sentence	Declarative Sentence
If you were my assistant, how would you plan my day?	Declarative Sentence	Declarative Sentence	Closed-ended Questions
What's your favourite colour?	Declarative Sentence	Declarative Sentence	Open-ended Questions

6 Discussion and Evaluation

The primary aim of this dissertation is to present a complete evaluation of three popular AI frameworks in the context of procedurally generating dialogues for Non-Player Characters (NPCs) in video games: OpenAI's GPT models, Convai, and Inworld AI. Two game prototypes have been developed with 3 of the proposed AI models integrated for non-Player characters to procedurally generate the dialogues. Evaluations have been done on these three models to determine the quality of the dialogues generated, performance of the models in terms of latency and the contextual relevance of the dialogues. The quantitative content analysis done for these three models shows the language richness, coherence, and level of involvement for these three models.

6.1.1 Quantitative Content Analysis (CQC)

As shown in the evaluation results, all three of these models can generate dialogues that are relevant for the topic of the conversation. Here's an in-depth discussion of various coding categories applied to evaluate these three models:

6.1.1.1 Topical Relevance

In a gaming context, the ability to engage players with relevant dialogues is essential for immersion. Except for one question ("As a merchant, what wares are you selling?"), all the queries show a high degree of topical relevance in Inworld AI. OpenAI and Inworld AI scored consistently high on topical relevance. ConvoAI, however, demonstrated a slight gap. High topical relevance is crucial for in-game dialogues to make sense in the context and keep the player engaged. OpenAI and Inworld AI have an advantage in this domain.

6.1.1.2 Informational Content

In terms of providing more informational content, OpenAI and Inworld AI once again surpassed ConvoAI. Rich informational content could enhance gameplay by offering depth and detail. ConvoAI might need improvements in this aspect to be more competitive.

6.1.1.3 Open/Closed Questions

OpenAI was balanced in responding to both types of questions. Inworld AI leaned more towards closed questions. ConvoAI on the other hand hardly asked any question. It leaned more towards declarative sentences. Versatility in addressing different question kinds is advantageous in a variety of game genres and settings. OpenAI has an obvious edge in this case.

6.1.1.4 Word Count and Sentence Length

When compared to ConvoAI and Inworld AI, OpenAI produces lengthier responses. Longer sentences can give rich content, but they must be well-paced to keep the action exciting.

6.1.1.5 Grammatical Errors

The sentences generated by all three algorithms were grammatically correct. Grammar is fundamental; thus, it's encouraging that all three models fared well in this category.

6.1.1.6 Coherence and Logical Flow

OpenAI and Inworld AI had better coherence and logical flow compared to ConvoAI. Coherence in dialogue contributes to a better narrative experience, an area where ConvoAI may require improvement.

6.1.1.7 Complexity

OpenAI and Inworld AI generated complicated sentences, whereas ConvoAI's interaction was simpler. A variety of complexity levels may be used to meet the needs of different players, but additional complexity often adds depth to the gaming experience.

6.1.1.8 Instructional Content

When compared to ConvoAI, OpenAI and Inworld AI performed better in generating instructional dialogues. Instructional content is essential for tutorial levels or missions, giving OpenAI and Inworld AI an advantage in these situations.

OpenAI appears to be the most well-rounded, excelling in most categories. Inworld AI follows closely but leans towards longer, more complex dialogues. ConvoAI, while competent, lags in informational content and complexity. Each has its own set of strengths and weaknesses, but OpenAI seems to be the most versatile and therefore best suited for a wide range of gaming applications. However, the specific needs of the game, such as the desired complexity or the type of questions primarily used, could tip the balance in favour of one of the other models.

6.1.2 Performance Analysis

The performance in terms of latency to generate the dialogue has been evaluated for each of these models. The speed at which a conversational model responds is critical in various applications, particularly in real-time gaming environments where delays can substantially affect the user experience. The mean response time for each of these models are obtained from this evaluation. The Mean Response Time for each of these models are:

- Inworld AI: 2.948 seconds
- Convai: 2.014153846 seconds
- Open AI: 16.93723077 seconds

This means that Inworld AI boasts a relatively fast response time, which could make it more suitable for high-paced gaming scenarios or tasks that require quick user interaction. With a mean response time of under 3 seconds, it can maintain the flow of conversation or game narrative without noticeable lag, contributing to a more immersive experience.

ConvoAI outperforms both InWorld AI and OpenAI in terms of response speed with a mean time of around 2 seconds. This could be crucial in applications where quick decision-making and responsiveness are imperative. The speedy response time could, however, be a reason for its slightly lower performance in some of the other categories like complexity and informational content. Quick but shallow responses may not always be ideal, depending on the application.

With a mean response time of 16.937 seconds, OpenAI takes considerably longer than its counterparts. This delay can potentially introduce lag in real-time applications, which might hamper user experience, especially in scenarios where swift feedback is crucial. While OpenAI's response time is noticeably longer, it may be offering depth, accuracy, and richness in its content. For real-time applications, Convai stands out as the most promising due to its speed. However, if the players allow for slightly longer waiting times and prioritizes detailed content, Inworld AI might be the go-to choose.

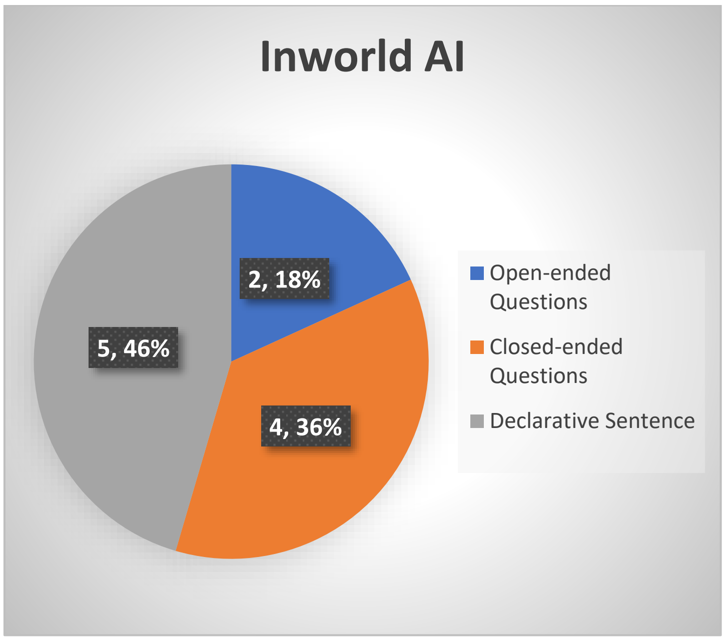
6.1.3 Contextual Analysis

The evaluation of coding categories across Inworld AI, Convai, and Open AI models reveals interesting differences in the way they handle various types of queries. These distinctions could be valuable for developers and researchers interested in employing these models in specific applications.

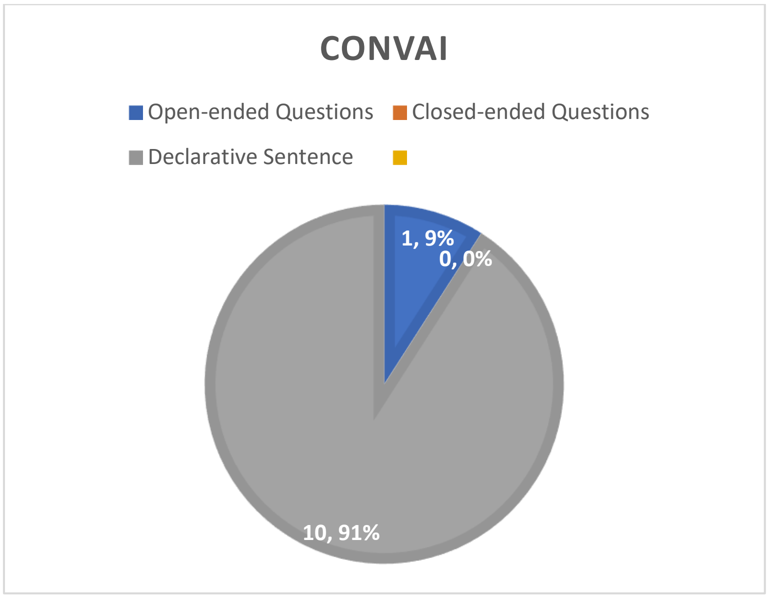
Understanding the Categories:

- Open-ended Questions: Queries that are designed to invite a full, meaningful answer using the subject's own knowledge and/or feelings.
- Closed-ended Questions: Questions that can be answered with "yes" or "no" or with a specific piece of information.
- Declarative Sentence: Sentences that provide information and can stand alone as a statement.

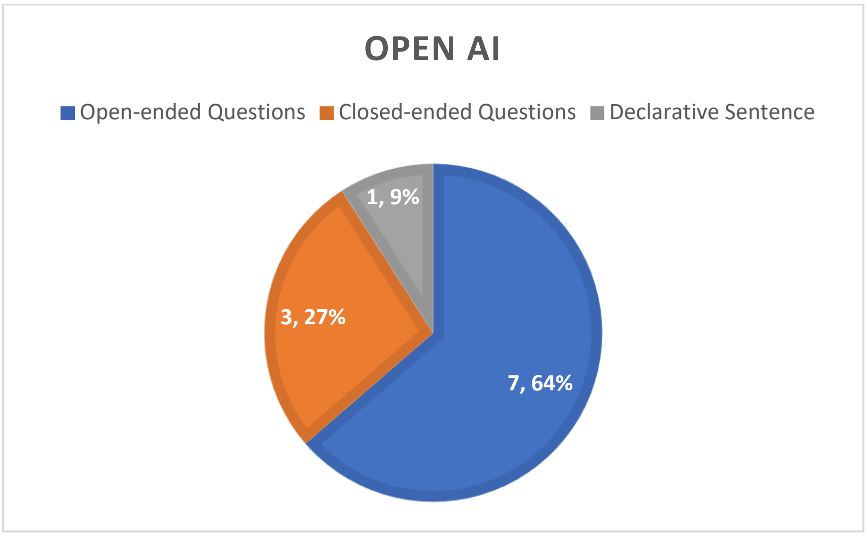
The following pie charts shows the contextual analysis of these three models:



IMG 6.1: Contextual Analysis of Inworld AI



IMG 6.2: Contextual analysis of Convai



IMG 6.3: Contextual analysis of Open AI

From this evaluation we can say that Inworld AI model seems to be balanced in responding to open-ended and closed-ended questions but leans more towards open-ended questions. It also tends to offer declarative sentences, which could be indicative of a more straightforward and informational approach to conversation. Unlike Inworld AI, Convai largely uses declarative sentences for most queries. This model seems geared towards providing straightforward information rather than engaging in open-ended conversations. This could make it less suitable for applications that require more exploratory or in-depth dialogues. OpenAI model prominently responds with open-ended questions and declarative sentences. This might suggest that OpenAI is more geared towards engaging the user in a dialogue that encourages deeper exploration of the subject at hand.

6.2 Evaluation and Reflection

The original software solution proposal envisioned a streamlined, highly efficient model comparison system capable of handling a wide range of discourse types and computational scenarios. However, as the project progressed, additional revisions were required. Initially, response time was not seen to be an important metric. However, preliminary experiments revealed that this parameter was critical for the practical use of these AI models in real-time gaming. This resulted in the addition of response time as a primary evaluation metric. The initial software was intended to compare only two models- OpenAI and Inworld AI. Later I figured out that it would be beneficial to compare several models, which necessitated significant refactoring of the initial software.

The decision to concentrate on OpenAI's GPT models, ConvoAI, and Inworld AI proved beneficial. These models represented a wide range of capabilities and allowed for a thorough comparison. The software was created in a modular manner, making it easier to incorporate additional features and metrics as the project progressed. The effort spent reviewing literature paid off by giving a solid foundation and grasp of the major metrics and evaluation approaches in the field of AI dialogue systems.

Each AI framework has its own set of APIs and configurations that required time to comprehend and integrate. Thorough documentation and online community support were extremely helpful in resolving these obstacles. In conclusion, the project was a learning experience with several obstacles that were generally handled through timely changes to the initial design, a focus on modularity, and an agile approach to problem-solving. The insights learnt on this voyage will surely assist future ventures.

7 Conclusions

The evaluation of advanced AI models from Inworld AI, Convai, and OpenAI in the context of procedurally generating NPC (Non-Player Character) dialogues has yielded a wealth of insights that provide not only an assessment of each model's capabilities but also their suitability for various types of interactive experiences. The mean response time becomes critical in an interactive setting such as video games, where NPC responsiveness can dramatically effect user experience. Convai excels in this area, with an average of little over 2 seconds, closely followed by Inworld AI. OpenAI's GPT models lag in this category, taking an average of 17 seconds to generate a response. This could be a bottleneck for real-time, fast-paced interactive applications. The assessment coding categories—open-ended inquiries, closed-ended questions, and declarative sentences—indicate the range and versatility of dialogues that these NPCs may generate. OpenAI preferred open-ended dialogues, which facilitated deeper interaction but may have increased response time. Inworld AI demonstrated balance by providing multiple types of statements and queries, allowing for more nuanced and contextually relevant interactions. Convai is primarily declarative, which makes it appropriate for specialised, uncomplicated tasks but somewhat less entertaining. OpenAI and Inworld AI frequently produced complex, information-rich responses, making them appropriate for instructional or narrative-heavy experiences. Convai's comments, on the other hand, were simpler and more straightforward, which may be more appropriate for casual or action-oriented games where language has a more basic functional purpose. Coherence and logical flow were generally high across all platforms, but OpenAI tended to edge out slightly in complexity and the ability to provide instructional guidance in talks.

It is important to note that this evaluation does not consider the simplicity of integration into game engines, nor does it consider the computational resources consumed by each model, which might be a key aspect for game developers. To summarise, the choice of AI dialogue system for procedurally produced NPCs is far from universal and should be heavily influenced by the type of interactive experience desired. To validate these findings, the following steps should include more detailed metrics and real-world application tests.

7.1 Future Work

The current research provides a basic grasp of the capabilities and limitations of various AI dialogue systems in the context of procedurally generated NPC dialogues. However, various potential study directions have been identified that could extend or enhance the work done thus far. Given that response time was a significant parameter in this study, future research should investigate ways to optimise response time for slower-performing models, including OpenAI's GPT models. While the current study concentrated on objective criteria such as response time and topical relevancy, future research could include human volunteers to provide qualitative insights into the user experience, dialogue immersion, and emotional engagement. Also, the computational cost of operating these AI models was not covered in this work. A full comparison of system resource use should be included in future work, as this is an important aspect for game developers. In addition, as AI becomes a more integral aspect of interactive experiences, ethical concerns such as data protection, content regulation, and fairness should be a focus of future study in this field. The findings could be validated with case studies where these dialogue systems are used in released games, thus showing their efficacy or limitations in real-world conditions. Future research can build on the findings of this dissertation by addressing these areas, providing a more comprehensive understanding of AI's involvement in crafting interactive narratives and experiences in gaming.

• References

Agis, R. A., Gottifredi, S. & Garcia, A. J., 2020. An event-driven behavior trees extension to facilitate non-player multi-agent coordination in video games. *Expert Systems with Applications,* 155(10).

Ahmad, M., Dennehy, D., Conboy, K. & Oivo, M., 2017. Kanban in Software Engineering: A Systematic Mapping Study. *Journal of Systems and Software,* Volume 137, pp. 96-113.

Ahmad, M. O., Dennehy, D., Conboy, K. & Oivo, M., 2017. Kanban in Software Engineering: A Systematic Mapping Study. *Journal of Systems and Software,* Volume 137, pp. 96-113.

Ahmad, M. O., Markkula, J. & Oivo, M., 2013. *Kanban in Software Development: A Systematic Literature Review,* s.l.: s.n.

AlbadarnehIsraa, A., Albadarneh, I. & Qusef, A., 2015. *Risk management in Agile software development: A comparative study.* s.l., s.n.

ALI, A., NAEEM, S., ANAM, S. & ZUBAIR, M., 2023. *Agile Software Development Processes Implementing Issues and Challenges with Scrum.* Basel, Switzerland, s.n.

Anderson, D. J., 2010. *Kanban: Successful Evolutionary Change for Your Technology Business.* s.l.:Blue Hole Press.

Anon., 2011. *The Application of AI for the Non Player Character in Computer Games.* s.l., s.n., pp. 1049-1050.

Anon., n.d. *One Piece Wiki.* [Online]
Available at: https://onepiece.fandom.com/wiki/Trafalgar_D._Water_Law

Arkoudas, K., 2023. *GPT-4 Can't Reason,* s.l.: Preprints.

Brown, T. et al., 2020. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems 33.*

Caroux, L., Isbister, K., Bigot, L. L. & Vibert, N., 2015. Player–video game interaction: A systematic review of current concepts. *Computers in Human Behavior,* Volume 48, pp. 366-381.

Cohe, J., P. C., West, S. G. & Aiken, L. S., 2002. *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences.* 3 ed. New York: s.n.

Conforto, E. C. et al., 2014. *Can Agile Project Management be Adopted by Industries Other Than Software Development?.* s.l.:Project Management Institute.

Conforto, E. C. et al., 2014. Can Agile Project Management be Adopted by Industries Other than Software Development?. *Project Management Journal,* 45(3), p. 21–34.

Cooper, R. G. 5. (. 2., 2016. Agile-Stage-Gate Hybrids: The Next Stage for Product Development. *Research-technology Management,* Volume 59, p. 21.

Davies, R., Dewell, N. & Harvey, C., 2021. *A framework for interactive, autonomous and semantic dialogue generation in games.* s.l., s.n.

De, B., 2017. API Management. In: *API Management.* Berkeley, CA: Apress, p. 15–28.

Denning, S., 2020. *Agile project management: The smart person's guide.* s.l.:TechRepublic.

Drageset, O., Winands, M. H. M., Gaina, R. D. & Perez-Liebana, D., 2019. *Optimising Level Generators for General Video Game AI.* London, s.n., pp. 1-8.

Gandomani, J. & Ziaei, M., n.d. An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach. *Journal of Systems and Software ,* Volume 107.

González-Sánchez, J. L. & Vela, F. L. G., 2014. Assessing the player interaction experiences based on Playability. *Entertainment Computing,* 5(4).

Henderson, P. et al., 2017. *Ethical Challenges in Data-Driven Dialogue Systems,* s.l.: s.n.

Hidalgo∗, E. S., 2019. *Adapting the scrum framework for agile project management in science: case study of a distributed research initiative,* s.l.: Heliyon.

Holtz, N., Wittfoth, S. & Gómez, J. M., 2023. *AI Meets Risk Management: A Literature Review on Methodologies and Application Fields.* s.l., IEEE, pp. 1-11.

Hoseini, D., 2023. *Dynamic Deviation Algorithm*, s.l.: s.n.

Hussain, M. Z., 2020. *Scrum Notes as Per Scrum Guide 2020.,* s.l.: s.n.

Jain, J., 2022. Testing the API. In: *Learn API Testing.* s.l.:s.n.

Jain, R., 1991. *The Art of Computer Systems Performance Analysis: Techniques For Experimental Design, Measurement, Simulation, and Modeling.* New York: Wiley.

Kerzner, H., 2018. *Project management best practices: Achieving global excellence.* s.l.:John Wiley & Sons.

Kerzner, H., 2022. *Project Management: A Systems Approach to Planning, Scheduling, and Controlling, 13th Edition.* s.l.:John Wiley & Sons.

Khan, S. M. A., 2023. *Waterfall Model Used in Software Development Reference: Software Requirements Engineering Waterfall Model,* s.l.: s.n.

Kim, G. et al., 2021. *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations.* IT Revolution: s.n.

Krippendorff, K., 2013. *Content Analysis. An Introduction to Its Methodology.* California: Sage Publications.

Kumar, A., Kumar, N., Mondal, S. & Biswas, T., 2022. *A Survey-Based Study to Understand Various Aspects of Kanban,* s.l.: s.n.

Kumar, Y. et al., 2023. *A Testing Framework for AI Linguistic Systems (testFAILS),* s.l.: s.n.

L'Esteve, R. C., 2023. Applying DevOps. In: *The Cloud Leader's Handbook.* s.l.:s.n.

Lab, Y. K., 2023. *A Testing Framework for AI Linguistic Systems (testFAILS),* s.l.: s.n.

Larson, E. W. & Gray, C. F., 2011. *Project Management: The Managerial Process,.* s.l.:McGraw-Hill Hills Companies.

Lazaridou, A., Hermann, K. M., Tuyls, K. & Clark, S., 2018. *Emergence of Linguistic Communication from Referential Games with Symbolic and Pixel Input,* s.l.: s.n.

McKenna, D., 2016. *The Art of Scrum.* s.l.:s.n.

Mehta, A., Kunjadiya, Y., Kulkarni, A. & Nagar, M., 2021. *Exploring the viability of Conversational AI for Non-Playable Characters: A comprehensive survey.* Mumbai, India, s.n., pp. 96-102.

Neuendorf, K. A. & Kumar, A., 2016. Content Analysis. In: *The International Encyclopedia of Political Communication.* s.l.:s.n.

Perez-Liebana, D. et al., 2019. *General Video Game AI: A Multitrack Framework for Evaluating Agents, Games, and Content Generation Algorithms.* s.l., s.n., pp. 195-214.

Pickett, G., Khosmood, F. & Fowler, A., 2021. *Automated Generation of Conversational Non Player Characters.* s.l., s.n.

Rabin, S., 2006. *AI Game Programming Wisdom 3 (Game Development Series).* Massachusetts: Charles River Media, Inc..

Radford, A., Narasimhan, K., Salimans, T. & Sutskever, I., 2018. *Improving Language Understanding by Generative Pre-Training,* s.l.: s.n.

Rawal, A. & Miikkulainen, R., 2018. *From Nodes to Networks: Evolving Recurrent Neural Networks,* s.l.: s.n.

Reiff, J. & Schlegel, D., 2022. Hybrid project management – a systematic literature review. *International Journal of Information Systems and Project Management,* 10(2), p. 45–63.

Ruane, E., Birhane, A. & Ventresque, A., 2019. *Conversational AI: Social and Ethical Considerations*, s.l.: s.n.

Rubin, K. S., 2012. *Essential Scrum: A Practical Guide to the Most Popular Agile Process.* s.l.:s.n.

Sarkadi, S., Tettamanzi, A. G. B. & Gandon, F., 2022. Interoperable AI: Evolutionary Race Toward Sustainable Knowledge Sharing. *IEEE Internet Computing,* 28 October, 26(6), pp. 25 - 32.

Scriven, P., 2023. A Social Phenomenology of Non-Player Characters (NPCs) in Videogames in advance. *Techné Research in Philosophy and Technology*, 27(2), pp. 240-259.

Serban, I. et al., 2015. A Survey of Available Corpora for Building Data-Driven Dialogue Systems. *ArXiv.*

Serrador, P. & Pinto, J. K., 2015. Does Agile work? — A Quantitative Analysis of Agile Project Success.. *International Journal of Project Management,* 33(5).

Smith, K., Miyashita, Y. & Sugawara, T., 2020. *Analysis of Coordination Structures of Partially Observing Cooperative Agents by Multi-Agent Deep Q-Learning.* Nagoya, Japan, Springer-Verlag, p. 150–164.

Sommerville, I., 2015. *Software Engineering.* 9 ed. s.l.:Pearson.

Stegeren, J. v. & Myśliwiec, J., 2021. *Fine-tuning GPT-2 on annotated RPG quests for NPC dialogue generation.* s.l., s.n., pp. 1-8.

Suyikno, D. A. & Setiawan, A., 2019. *Feasible NPC Hiding Behaviour using Goal Oriented Action Planning in case of Hide-and-Seek 3D Game Simulation.* s.l., s.n.

Tausczik, Y. R. & Pennebaker, J. W., 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology,* 29(1), pp. 24-54.

Tian, J. et al., 2020. *User Intention Recognition and Requirement Elicitation Method for Conversational AI Services,* s.l.: s.n.

Togelius, J. & Yannakakis, G. N., 2016. *General general game AI.* s.l., s.n., pp. 1-8.

Uzwyshyn, R., 2023. *Can We Get Some Agile Here? The Application of Agile Project Management Principles for Library IT,* s.l.: International Federation of Library Associations and Institutions (IFLA).

Vaswani, A. et al., 2017. *Attention is All you Need.* Long Beach, CA, USA, s.n.

Vial, G., Cameron, A.-F., Giannelia, T. & Jiang, J., 2022. Managing artificial intelligence projects: Key insights from an AI consulting firm. *Information Systems Journal,* 33(3), p. 669–691.

Warpefelt, H., 2016. *The Non-Player Character: Exploring the believability of NPC presentation and behavior,* s.l.: s.n.

Whiting, E. & Datta, S., 2021. *Performance Testing and Agile Software Development: A Systematic Review,* s.l.: s.n.

Yin, R. K., 2017. *Case Study Research and Applications: Design and Methods.* s.l.:SAGE Publications, Inc.