CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Applying ChatGPT in AI-based dynamic video game narrative generation system

Master's thesis in Computer Science and Engineering

Artūrs Umbraško and Robert Drury

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

Applying ChatGPT in AI-based dynamic video game narrative generation system

Artūrs Umbraško and Robert Drury

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

Applying ChatGPT in AI-based dynamic video game narrative generation system
Artūrs Umbraško and Robert Drury

Supervisor: Pauline Belford, Department of Computer Science and Engineering
Examiner: Mikael Wiberg, Department of Computer Science and Engineering

Master's Thesis 2023
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2023

Master's thesis in Computer Science and Engineering
Artūrs Umbraško and Robert Drury
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

ChatGPT's release and general availability has the potential to create new possibilities for storytelling in games. There are potential benefits of leveraging ChatGPT to generate a narrative at runtime in a video game, as well as possible resulting pitfalls. To explore these factors, we created a game prototype that dynamically instantiates game assets from JSON, generated by ChatGPT. We additionally incorporate a dynamic status-effect system and, in conjunction with textual player input, utilise this in order to instruct ChatGPT to generate an informed narrative as the game is played. Player feedback indicates that ChatGPT's addition is beneficial overall, though not without some downsides.

Keywords: Computer, science, engineering, thesis, ChatGPT, OpenAI, games, narrative, machine learning, interaction design

Acknowledgements

Thanks to Pauline for her supervision and invaluable advice, Laure for her guidance and support, Sammy for teaching me the meaning of dogged insistence, and to all of our playtesters for their feedback and time.

Robert Drury, Gothenburg, May 2023

Thanks to Pauline for making this thesis possible by being our supervisor and teacher, to Daniella for her unwavering support and encouragement, to my friends and family who were by my side during this time, and to our playtesters for volunteering and providing insightful feedback.

Artūrs Umbraško, Gothenburg, May 2023

Contents

List of Figures

List of Tables

1

Introduction

Artificial Intelligence is nowadays as popular as it has ever been. New advancements in the conversational AI offered by ChatGPT [1] and AI Dungeon [2] suggest new opportunities for the dynamic creation of content in video games. At the same time, GitHub Copilot allows the developers to be more efficient at their job through automated code generation. However, game developers still lack tools that will allow them to use this new technology to create game elements and game environments more efficiently and quickly [3], and minimise the repetitive and manual tasks that are usually a part of the world-building process.

ChatGPT is an Artificial Intelligence chatbot, developed by OpenAI[4]. It is based on generative pre-trained transformer, a type of Large Language Model. Language models are statistical models which are capable of performing natural langauge processing tasks by assigning probabilities to sequences of words. With the usage of large datasets to train these models, Large Language Models have the ability to generate fluent natural language [5].

Tools like AI Dungeon and ChatGPT are limited to conversational text. We propose that their ability to partially understand text and context can result in applications of the technology beyond mere text - video game elements can be dynamically created and modified. The range of possible scenarios a game can represent can be expanded by leveraging the tools' ability to parse natural language in order to compose appropriate, dynamic settings and game elements. This would also allow for more emergent narratives.

1.1 Research Question

In summary, and in the context of the prototype we made while writing this thesis, we have been concerned with the following research questions:

1. What are the constraints and affordances of incorporating ChatGPT into a simple turn-based strategy/role-playing game, for the purpose of dynamically creating an in-game narrative based on player input and in-game context?
2. Does incorporating ChatGPT into such a project have a positive influence on the player experience?

Both of these questions can also be summarised simultaneously with the following research question: **Do the benefits of incorporating ChatGPT into a simple turn-based strategy/role-playing game exceed the disadvantages and challenges of doing so?**

Our goal was largely to come up with ways to leverage AI and make dynamic content, based on text that has a bearing on in-game events. Given that tools like ChatGPT can interpret text and gain some apparent understanding of context, we believe that the generated text can be used as input for a game, and serve as a basis to dynamically instantiate and modify assets in real-time. This enables the text-based tool to be both storyteller and facilitator of the creation of the assets.

Over the course of our prototyping work[1], we became more selective about the ways in which we integrated ChatGPT as we discovered its shortcomings and capabilities, as well as making some judgements about how best to leverage it for our particular prototype. Our approach uses a combination of ChatGPT's ability to output JSON as a means of instantiating enemies in our game, and its ability to produce small narrative passages based on input created by us and supplemented with both in-game textual player input and in-game player action. This is facilitated by an API hosted by OpenAI, which we access at particular times during gameplay.

[1]The prototype can be downloaded here: `https://github.com/necrosmash/thesis_prototype/releases/tag/v1.0.0`

2

Academic Context

2.1 Artificial Intelligence in Video Games

Perhaps, the most common way of creating artificial intelligence is through finite state machines. Finite state machines are mathematical models that use a finite set of *states*, and inputs are used for the transition from one state to another [6]. This technique was used long before any kind of machine learning was adopted in video games.

Another approach, and the one we are most curious about for the sake of this thesis, is machine learning. Machine learning is a subfield of artificial intelligence that involves the development of algorithms and statistical models that enable a system to learn from and make predictions or decisions without being explicitly programmed to perform a specific task [7]. In contrast to finite state machines, it is not bound by a finite set of possible states, and therefore can be used to produce more flexible results. Neural networks are a subset of machine learning that is specifically concerned with creating algorithms that simulate neurons to complete such tasks.

Recent publications suggest that most of the algorithms are not utilising machine learning as a part of their content generation. Freiknecht and Effelsberg [8] conducted a thorough review of which algorithms are used for procedural content generation in various video games. It seems that the methods used are not relying on machine learning, and in this paper we aim to see how this status quo can be challenged.

2.2 Procedural Content Generation

Creating game content automatically is something that has been previously done many times before. Both current and previous generations of games have employed Procedural Content Generation to accomplish this goal. The advantages of this method include greater playability of games and more variation in the game content that the player may encounter. The type of game content that can be generated automatically varies greatly from game to game. [9] includes a taxonomy of procedurally generated game content. The taxonomy includes game content from the atomic

game assets all the way to what it calls "derived content", or "metagame". The paper goes in-depth into each specific taxon, but in short, these can be summarised as:

1. Game Bits are the elementary units of game content, the *game assets*.
2. Game Space is the environment in which the game takes place, which is filled with Game Bits.
3. Game Systems are the complex systems and models of the game, used to simulate complex environments.
4. Game Scenarios are the scenarios related to the game events, including levels and story.
5. Game Design includes the ruleset, including what the player can do in the game, as well as the goal of the game.
6. Derived Content is defined as content derived from the game, side-products of the game.

The Game Space and Game Scenarios are the most relevant for this report. Game Bits generation has been discussed thoroughly in papers before. We are interested in exploring the generation of Game Space through the manipulation of Game Bits, and Game Scenarios as a result.

[9] uses this taxonomy and builds upon it to discuss modern trends within this subject. Notably, Bontchev concludes by highlighting that novel methods of Procedural Content Generation are to be applied by designers in order to meet the players' demand for fresh game content.

Various techniques and algorithms have been employed in the field of Procedural Content Generation to generate content. [9] provides the following classification of these methods:

1. Pseudo-random number generators
2. Generative grammars—Lindenmayer-systems, split grammars, wall grammars, and shape grammars
3. Image filtering—binary morphology and convolution filters
4. Spatial algorithms—tiling and layering, grid subdivision, vectorization, fractals, and Voronoi diagrams
5. Modeling and simulation of complex systems—cellular automata, tensor fields, agent-based simulation, and other complex systems and theories
6. Artificial Intelligence (AI)—genetic algorithms, artificial neural networks, and constraint satisfaction and planning based on using PDDL, i.e., Planning Domain Definition Language [10]

[11] goes in-depth into methods based on machine learning used in Procedural Content Generation. Summerville et al. select Frequency Counting, Backpropagation, Evolution, Matrix Factorisation and Expectation Maximisation, and demonstrate how these methods can be applied to generate video game levels. [12] proposes a model, describing the roles that Procedural Content Generation via Machine Learning can assume in relation to the player (Table 2.1).

Pattern	What player(s) do	Role of AI (in relation to player)	Example(s)
AI is Visualized	Observe AI state	Gives (strategic) information, showing states	Third Eye Crime
AI as Role-model	Imitate AI	Show agent actionas and behaviors, agents as puzzles	Spy Party
AI as Trainee	Teach AI	Child/student	Black & White
AI is Editable	Edit AI	Artifact/agent that player can author/manipulate	Galactic Arms Race
AI is Guided	Guide/manage the AI	Partly independent inhabitants, with players as their Gods	The Sims
AI as Co-creator	Make artifacts assisted by AI	Co-creator, making artifacts	ViewPoints AI
AI as Adversary	Play game against the opponent	Opponent (symmetric)	Chess, Go
AI as Villain	Combat the Villain(s)	Villain in game; mob, boss mob, NPC (asymmetric)	Alien Isolation
AI as Spectacle	Observe	Spectacle, enacting simulated society	Nowhere

Table 2.1: AI Patterns and Roles in Video Games [12]

2.3 AI-Based Game Design

There have been discussions about how AI will shape games in the future, and how AI can and should be leveraged in order to do so. Eladhari et al. [3] argue that AI Design and Game Design should both be considered when leveraging AI to make a game, and that they should each affect the other. While this paper lists some examples of pre-existing games, it also argues that the space requires further exploration:

> We argue that the development of innovative artificial intelligence (AI) systems plays a crucial role in the exploration of currently unreachable spaces. To aid in exploration, we suggest a practice called AI-based game design, an iterative design process that deeply integrates the affordances of an AI system within the context of game design. [3], pg. 1

In [12], various roles are discussed that the AI can take in video games. The paper describes these roles and uses examples of video games. However, most of these examples use AI that is based on finite-state machines, as opposed to the other methods mentioned above: that is machine learning and neural networks. Exploration of what is possible if some or all of these roles, or a similar approach, are assigned to a machine learning agent can provide us with important insights into how this technology can be used as a part of designing games. We are interested in studying how modern machine learning technology can be used to improve content generated through finite state machines and similar more rigid algorithms.

Outside of the examples provided in [12], there are other notable attempts at AI-based gameplay in the literature. [13] describes an attempt to create a gameplay

based on AI as Trainee. In the game, the player is tasked with teaching AI *minions* to move boxes around the level. The *minions* are learning through trying to replicate the actions of a player. The player can temporarily stop the *minions* from tracking player movement, which means that the *minions* will no longer learn from player actions, until the the recording is turned on again. This is a part of the main game mechanic, and can be used by the player to avoid teaching *bad habits* to the *minions*. As one of the conclusions, the paper highlights the fact that the *minions* in the game turned out to be disappointing due to their perceived *intelligence* levels flattening out pretty quickly:

> At the same time, the zombox theme and game design already tries to balance that out with the Player vs. Player mechanic, but it overall can't be helped that players might look at their running boxes with the look and feeling of disappointed parents. [13], pg. 398

2.4 AI-Based Narrative in Video Games

Narrative-focused research relating to AI-based games has been done before, but since the types of tools offered by OpenAI (e.g. ChatGPT) did not exist then, they rely on other ways of tracking / generating narrative (or forego solid narratives entirely). Matteas [14] refers to "subjective avatars" as a way of creating characters that have their own subjective state - a narrative exists and is formed elsewhere, and the subjective avatars (implemented using Em [15] and Hap [16]) are used to interpret this narrative and supplement it with their own feelings and opinions.

Perceptual Experience Management is an approach to "mediating between player choice and author constraints" [17] - it aims to provide a systematic way to reconcile "authorial control" and "player choice", and does this by making use of a tool called the General Mediation Engine [18] (GME).

GME is a "plan-based interactive narrative generation process". GME takes, as input, a PDDL (Planning Domain Definition Language) domain and problem file. The domain file specifies "action operators" - these are things that can be instantiated in the game world - whereas the "problem file" specifies the initial state of the world, as well as its goal state. GME is also capable of outputting English sentences to describe game scenes, instead of simply displaying the raw, underlying state to the player - this is what is done in Robertson and Young's paper [18]. This approach also offers an interesting way of demonstrating GME's capabilities - by demonstrating a text-based game for the purposes of the paper, Robertson and Young highlight GME's ability to be separated from the game engine itself. This means that GME shares a lot in common with our approach - there is a clear delineation between the text-based narrative and the actual game assets that the player is interacting with. The difference is that, with GME, the text displayed to the user is a by-product of the underlying state that GME is tracking. As we use OpenAI's tools to make a narrative for us, and make game scenes from that narrative, we are taking a different approach - instead of just generating descriptive text from underlying game state, we are also instantiating game assets from ChatGPT's responses.

GME makes use of a "planner", of which Glaive [19] and Fast Downward [20] are two possibilities. Fast Downward was used by Robertson and Young for their paper [18], but another possibility is Glaive, which has been used to generate "disaster storylines" [21]. Given how specific the topic is, the authors decided to use a "typhoon emergency plan manual" to build an appropriate domain model. This is an interesting approach, but is quite specific - we believe that by using OpenAI's tools one can generate a greater range of possible narratives, all without needing to create a domain model in the first place.

Glaive has also been used with Stories of the Town [22] to generate narratives. In this paper, Talk of the Town [23] (a simulation of fictional characters in a simulated town) is used in conjunction with Glaive to generative narratives. This paper is only concerned with generating narratives, and does not concern itself so much with any potential game engine using the narrative under the hood. However, the authors have created a system that is capable of producing a complete, readable narrative.

Prom week [24] is a game worth mentioning in relation to interactive narratives. It is a game that was developed in addition to an AI system called "Comme il Faut" (French for "properly") (CiF) [25]. The reasoning behind developing this system is described as one that attempts to overcome many of the issues in pre-existing narrative-containing games. Examples given include Mass Effect [26], a game series which limits meaningful conversations between characters to specific locations at specific times in the games' campaigns; Star Wars: The Old Republic [27], a game whose development made use of a large team of full-time writers for years before the game entered the pre-production stage, thereby exemplifying a development strategy inappropriate for many development teams; and The Sims [28], a game which, despite giving players a lot of control to "develop a social space between the characters", contains very abstract representations of narrative. As such, the team that developed Prom Week and CiF attempted to create a game that more-closely connects gameplay and story:

> Because the gameplay of Prom Week involves manipulating the social space, which is the primary story content of the kind of high school narrative we wanted to emulate, the gameplay is the story. Every action the player takes advances the game's narrative and sends ripples throughout the internal social state, which in turn affects which actions are available in subsequent turns. The system is a partner of the player, giving the narrative meaning and shape. This is in contrast to a sandbox game in which gameplay may be the story, but the story is formed only in the mind of the player, and not understood or reasoned over by the system. [25], pg. 3

Prom week is a game that concerns itself with the social lives of its characters. Each "story" consists of the player attempting to guide the characters in order to fulfil a goal for a given character. Multiple approaches can be taken to attain the character's goal - for example, if an unpopular character wants to befriend a popular character, one possible solution would be to make the unpopular character more

popular; another solution would be to make the popular character less popular, in order to remove their inhibitions from befriending someone who isn't popular. Players initiate "social exchanges" between the characters in order to achieve their goals, and changes to the social state are managed by CiF.

1001 nights [29] is a game whose setting is heavily-inspired by One Thousand And One Nights [30]. It uses a combination of a trained GPT-2 model with input data from the Guthenburg Short Story collection [31], and dreamily.ai [32] to produce a game about a woman and a king collaborating in order to tell a story. After telling part of a story, the second phase of the game consists of a battle in which the woman can use the previously-mentioned weapons and armour in a battle against the king. Thematically, the idea is that this woman has the ability to conjure physical implements from words.

While this game is mostly text-based (until the battle phase), the parsing of nouns in order to create physical objects is particularly noteworthy for our purposes. However, the game is very limited with respect to the types of objects that can be created - swords and shields are essentially all that can be used in the battle phase, and there are only a few synonyms at play - "sword", "knife", "blade", and "dagger" all map to the same physical item for combat purposes, as do "shield" and "armour". The authors recognise these limitations, and state that one possible way of rectifying this is to use WebVectors [33], permitting the use of a greater range of words for their purposes.

2.5 OpenAI and ChatGPT usage

ChatGPT is still relatively new, and at the time of writing people are still discovering new uses for it. These uses hint at OpenAI's APIs usage as a backend in games, which we are aiming to accomplish. Examples of ChatGPT's capabilities include:

- Fixing bugs in code, and explaining the fixes [34]
- The creation of web scraping tools from scratch [35]
- Collaborate on creative writing tasks [36]
- Function as a database [37]
- Generate WordPress plugins [38]

Following on from these earlier showcases of its capabilities, Microsoft incorporated ChatGPT into its Bing search engine [39].

In the absence of an official API for general usage, ChatGPT's usage in actual applications remained sparse for a time. In the earlier stages of our thesis work, an official API became available (though official APIs for other models, such as its DaVinci model, were already available). Perhaps its most pre-existing interesting use (in relation to our purposes) is as the facilitator of play sessions of Dungeons and Dragons [40], as Dungeons and Dragons exemplifies a game whose range of narrative possibilities eclipses most video games, thanks to its reliance on player input for

telling a story. We are particularly interested in how ChatGPT can be leveraged for narrative purposes, given its ability to receive text as input and generate appropriate text as output.

2.6 Summary

Based on the previous discussion, we can see that Procedural Content Generation has been successfully used to provide the players with complex, replayable experiences. A lot of the effort went into Procedural Generation of game assets and game environments. While examples like [24] exist, that prove that it is possible to create narrative gameplay using AI systems, we can see that these attempts have their own constraints. In the particular case of Prom Week, the thematic constraints are created by the game developer, and the AI system is mostly concerned with regulating the interactions between characters.

While taking inspiration from previous examples of Procedural Content Generation, especially in narrative-heavy examples, such as Prom Week, the aim of this thesis is to explore the opportunities that novel AI technology can offer to game developers to create emerging, interactive narrative-based player experiences.

In summary, while the usage of AI tools in video games is nothing new, we have found that there is unexplored territory in this space, especially now that tools like ChatGPT are available and easily accessible. Much use of AI tools is not concerned with generating narrative, and while Prom Week offers interesting insight into the kinds of games that are possible, it also uses specialised tools that were developed in conjunction with the game. We are interested in using ChatGPT, as it is a more general and publicly accessible tool, with a lower barrier to entry.

3

Methodology

Finding the solution to the research questions outlined in this thesis requires researching various related topics, such as the aspects of a technical implementation of an AI-based narrative gameplay system, as well as practical work that includes the design and prototyping of an artefact. After the development of an artefact, the aforementioned artefact needs to also be evaluated with real users. Therefore, this study can be divided into the following steps:

1. Research
2. Prototype Design
3. Prototyping
4. Prototype Evaluation
5. Results and Discussion
6. Conclusion and Report Writing

3.1 Scrum methodology

When working on the project, we followed a Scrum philosophy of working in two-week sprints in which vertical slices of functionality will be delivered. Scrum is an Agile Method in which projects are split into "sprints" consisting of a prioritised backlog of tasks [41].

Before commencing a two-week sprint, work was itemised into a backlog of items whose highest-priority constituents were at the top. After completing a sprint, we evaluated what we had accomplished and decided if we wanted to begin work on a new prototype or continue adding functionality to the old prototype. This process is visualised in Figure 3.1. We wanted our prototyping work to utilise a combination of throwaway prototyping and iterative prototyping, with decisions made frequently on which strategy to employ for future work.

3.2 Prototyping

Initially, a throwaway method of prototyping was proposed. It included that prototype design, prototyping, and prototype evaluation steps of the study be repeated. As we had discussed at the beginning of the project, this would allow us

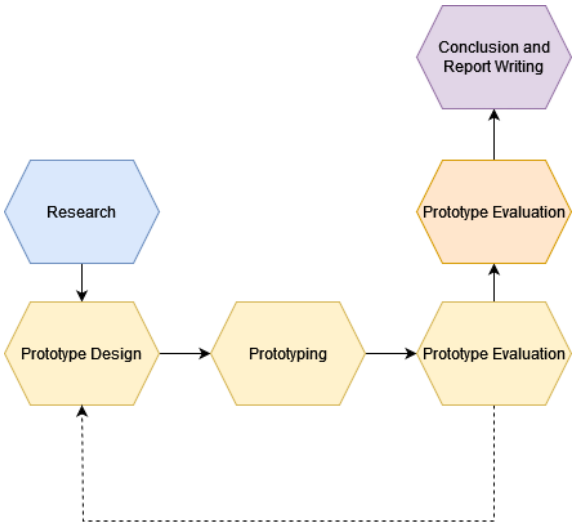


Figure 3.1: The project development flow

to create multiple playable demos of games, that would employ dynamic AI-based narrative. Evaluating these demos would gain us insight into more possible uses of AI in video game narrative at once.

Throwaway prototyping is a form of prototyping in which a prototype is developed and then discarded [42]. Since our plan was originally to make multiple prototypes, we foresaw the need to discard prototypes and start afresh, using what we had learned from the previous prototype as guidance for the next prototype's design and implementation.

Incremental prototyping, contrasting from throwaway prototyping, is a form of prototyping in which different prototypes are combined into one "product" [43]. We wanted, from the beginning, to use this type of prototyping where appropriate. We decided from early on that if we completed a sprint and believed that we could still learn more by continuing development on the current prototype we would create a new increment of the prototype rather than discarding it at that point in time.

As our first sprint approached the end of its cycle, we had implemented a very basic vertical slice of a game. It included a simple map generation, combat and story generation systems. As such, we had decided to continue working on the same prototype, instead of looping back to the design phase in order to work on a new demo. It was the first instance when we decided to follow incremental prototyping in favour of throwaway prototyping. The reasoning behind this approach was that after internal playtesting we had discovered that there was simply not enough gameplay to evaluate, so if we proceed with evaluating the prototype and building a new one, any response that we would have received from the users would most likely insufficient, since it would be hard for them to evaluate such a limited implementation of the concept we were trying to demonstrate.

Later, as we were closing our second sprint, we had a team discussion about where to proceed with that. A lot of new features were added, but we still considered

that the amount of gameplay that we had in our prototype was not enough to start evaluating it with the users. We had decided to continue our third incremental prototyping sprint on the same prototype. Since this also meant that when that sprint would be over, we would have very little time to develop a second prototype, and we would likely face the same issues with lack of content afterwards, it was decided that we would not use throwaway prototyping, and would instead focus on incremental prototyping only, and would continue working on the same demo afterwards.

3.3 Playtesting methodology

There are a number of methods that can be used to conduct playtesting, and for this project, we decided to evaluate using a combination of one-on-one testing, feedback forms and an open discussion.

One-on-one testing involves a playtesting session with only the playtester and the game master. The playtester is asked question both before and after the play session, and during the play session, the game master watches the playtester play the game and takes notes. A contrasting method is group testing, where there are more than one playtester. This can be beneficial in some contexts, but may also create an unwanted group dynamic among the playtesters.

Feedback forms provide an opportunity for the game master to effectively collect quantified data. Each playtester is provided with a standartised list of questions to answer, which makes it easier to compare the data afterwards. This can be done both on paper and digitally, through services such as Google Forms.

Open discussion is a playtesting method that involves a discussion of the game experience after the play session, either between the playtester and game master in case of a one-on-one testing, or between a group of playtesters and the game master. The discussion itself can be either more free or more structured. In our case, the discussion was more structured, since we had a specific area of interest that we wanted to discuss with the playtesters. During the discussion, the game master would take notes.

An additional technique that we used in our playtesting is what Fullerton calls "Do Not Lead" [44]. This means that the playtesters are observed without intervention, without telling them what to do. This would allow us to identify aspect of the game that the players have most issues with, as well as observe how they would naturally interact with it.

Another possible method of gaining large amounts of quantitative data are metrics. These include distributing the game to the players and letting them play the game normally, while also using special tools that allow to collect data about how the players play the game, such as their behaviour or what features are used more or less. However, this requires a lot of participants, additional development time, and a game that is or is close to a finished product.

3.4 Evaluation

After the prototyping phase, the project enters the evaluation phase. A test prototype will be provided to the users, and they will be able to interact with the game. This would allow us to gather data about how the participants experienced the game, and we can use this data to discuss and draw conclusions to our research question.

Based on the methodology discussed previously, a playtesting protocol was designed. The protocol in full can be found in Appendix E. In short, the main element of our evaluation was a playtesting session [44]. The playtesting was conducted online. A playtesting session consists of the following steps:

1. **Introduction**
   The playtester is introduced to who we are, what we are doing, and receives a brief explanation of the playtesting process. The playtesters are also informed that the playtesting sessions are being recorded.
2. **Warm-up discussion**
   The playtester answers a set of questions regarding the type of games they usually play.
3. **Play session**
   This is the step where the playtesters get to play the game. Playtesters play the game for about 20 minutes, while the game master is watching them. Playtesters are also asked to think out loud, in order to get more information about their expectations.
4. **Discussion of Game Experience**
   During this step, the playtesters answer a set of questions about things like overall appeal, interest level, and others. Additionally, the playtesters are asked questions that focus on the most important design questions at the given moment in the design process.
5. **Wrap-up**
   This is the shortest step, where the game master thanks the playtesters for their participation.

4

Planning

Based on the steps from the previous section, the schedule for this project was divided into weeks, and each week was assigned to one of the five blocks:

- Literature research - a block dedicated to research and analysis of literature relevant to this project
- Design - a block dedicated to design of an artefact
- Prototyping - a block dedicated to development of an artefact
- Evaluation - a block dedicated to playtesting or other evaluation method for the artefact
- Report writing - a block dedicated to the writing of this paper

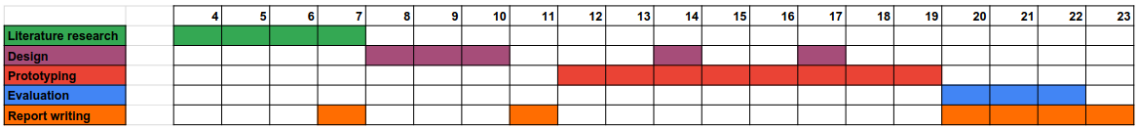


Figure 4.1: The proposed weekly schedule for this project

5

Prototyping

Initially, our vision was to create multiple small prototypes as proofs of concept, that could later on be expanded upon with additional mechanics and features. Our plan was to create multiple prototypes and use a combination of throwaway prototyping and incremental prototyping. Throwaway prototyping was appealing to use because it would allow us to experiment with different types of game and explore how ChatGPT's integration could uniquely benefit each of these prototypes. However, in the end, we focused all of our efforts on one prototype through incremental prototyping. There are two interrelated reasons for this. The first is that the time constraints made it difficult to make multiple prototypes. Secondly, we needed a game prototype that was substantial enough so as to be worthwhile to integrate ChatGPT into in the first place. If we were to make multiple prototypes, we ran the risk of creating games that were so basic that ChatGPT's integration did little to nothing to enhance the experience of playing the prototype, since there wouldn't have been much substance to the prototype in the first place.

The prototype we went on to create consists of a simple level, and uses ChatGPT to create dynamic, non-deterministic events that happen in response to the player using the game mechanics and, at set times, entering text.

5.1 The Design of the Prototype

We wanted to make a game that was sufficiently simple to both create and play. The simplicity gave us two main advantages - firstly, with the time constraints that we had, we didn't want to create something overly-ambitious that would cause us to run out of time during the course of our thesis work. Secondly, a simple design allowed us to keep our options open with respect to what kind of narrative we would create with ChatGPT, and how much that narrative would be affected by the internal state of the game.

We experimented with paper prototypes (Figure 5.2) in order to come up with an overall game design we were happy with. This permitted rapid iteration and an extremely low time investment. Once we had agreed upon the basic rules, we made a basic mock-up of what we wanted the design to be. This concept can be seen in Figure 5.1.

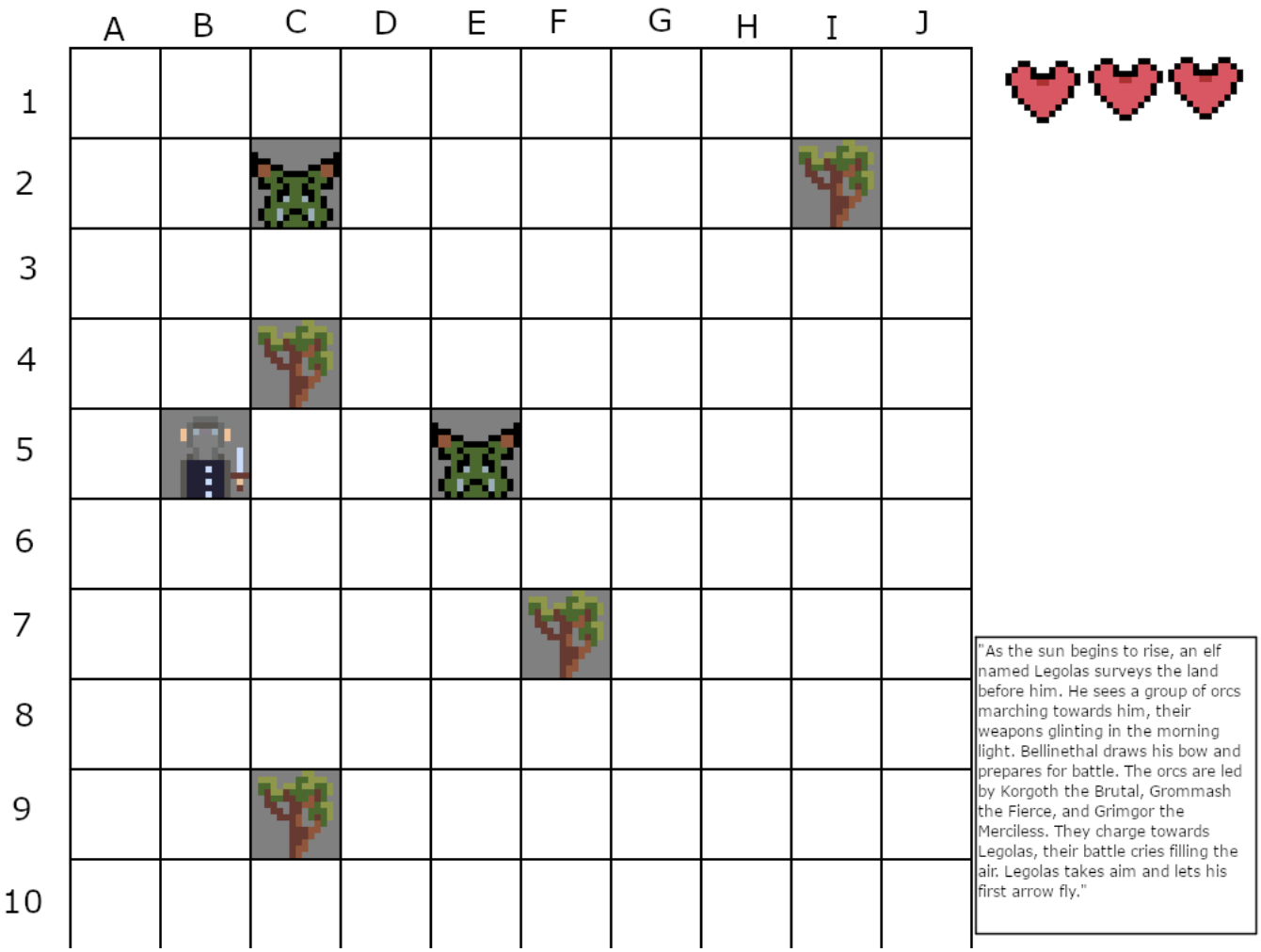


Figure 5.1: The design of our prototype

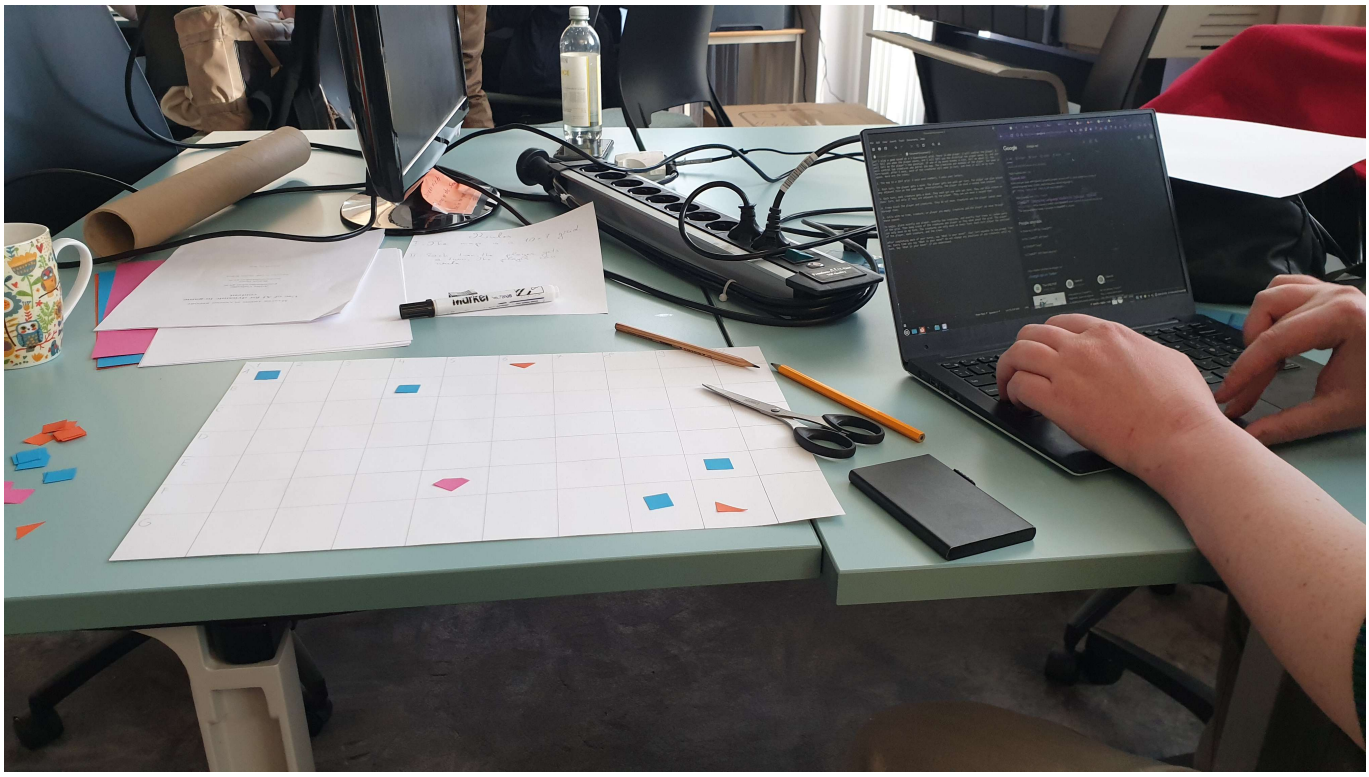


Figure 5.2: Paper prototyping session

The game is a top-down, turn-based game in which the player and patrolling enemies ("orcs") move around on a grid. Enemies have patrol points that they aim to arrive at. Upon arriving at their patrol point, enemies return back to their starting point. This process repeats until the enemy "spots" the player (the "elf"), at which point their behaviour changes - they will now chase the player until either they are defeated, or the player is defeated.

The basic structure of this game consists of:

- One player character
- Three to four enemy creatures (orcs), with the properties of these creatures, such as their size, the weapons they wield, and some "traits" generated at runtime through AI-based backend
- One or more obstacles that prevent the player and the creatures from travelling through them
- A grid-based two dimensional level consisting of cells, where each cell can either be empty, or contain a creature or obstacle (never both)
- A dynamic, AI-based system of generating this map, as well as additional flavour text at runtime
- A simple set of game rules and mechanics: turns, cell-based map traversing, combat and health, and game over conditions

Obstacles also exist on the grid, and obscure the vision of the patrolling enemies. Obstacles can be either trees, rocks, or explosive barrels. Explosive barrels can be

struck by the player, causing the barrel to explode, after a delay. When the barrel explodes, damage is dealt to any creature (player or enemy) in the barrel's proximity.

Enemies can be armed with either melee weapons (axe, hammer, sword) or ranged weapons (bow). The aim of the game is to defeat all of the enemies on the grid.

A scrollable text log shows the story, as it progresses. It is also used as a sort of "combat log" - when a creature takes or deals damage, a barrel explodes, an enemy spots the player, or an enemy is defeated, the log will record the event. The idea is to combine both the ongoing narrative and the immediate happenings in the game in a single, scrollable block of text.

5.2 Technology

The game prototype itself is a Unity project. This engine was chosen for numerous reasons:

- Our familiarity with Unity development
- The large amount of online documentation and discussion that aids development
- The popularity of the engine, which increases the possibility of tools necessary to communicate with external APIs being ported to Unity

While some external tools for communicating with OpenAI APIs exist [45], we found incorporating them into our prototype to be burdensome, and were ultimately unsuccessful. Following on from this, we attempted to incorporate a more-generic REST client [46], but came across similar issues with the version of Unity (2022.2.8f1) we were using. As a result, we implemented our own REST client, using Unity's own UnityWebRequest [47] functionality.

OpenAI [4] (the company behind ChatGPT) provides APIs for using some of its GPT-3 models. We use such an API as a backend for our prototype. Our plan was initially to use OpenAI's DaVinci model as, when work was commencing on the prototype, this was the most capable model that OpenAI hosted an official API for. However, during the earlier work of the prototype, a ChatGPT API became available [48]. Since it seemed more capable, we opted to use it instead of DaVinci.

The model is utilised by invoking REST API methods that OpenAI exposes for use. This permits us to create both a game scene and a narrative by simply conversing with the ChatGPT model through the API. At the beginning of the game, a network request is sent to the ChatGPT API (Appendix D) in which we specifically request of ChatGPT that it responds with a JSON object. This JSON object contains the following properties:

- `openingScene`: A string for containing the narrative opening scene that ChatGPT returns
- `orcs`: An array of `Orc`, which is a type for containing information about an

enemy

`Orc` contains a series of properties that ChatGPT decided the initialising values for. These properties are:

- `name`: The orc's name
- `description`: A description of the orc (often explaining something about their appearance)
- `weapon`: Can be `sword`, `hammer`, or `bow`
- `size`: Can be `small`, `medium`, or `large`

This JSON object is serialized to a class we call `BattleInfo`. `BattleInfo` contains all of the data we need to begin the game - enemies are instantiated and the narrative is displayed on the right-hand side of the screen.

Before beginning our prototyping work, we foresaw the need to settle on an agreed-upon degree of variation for the words used to instantiate our assets. For example, "melee weapon" could easily be extrapolated out into a wide range of possible objects - melee weapons could include hammers, axes, swords, maces, clubs, spears, halberds, glaives, and many others. This was a question of how to apply the technology at our disposal. As was previously mentioned WebVectors [33] offers one possible way of taking a range of possible terms and mapping them to a narrower range of instantiable assets. However, another possibility, and the one we ultimately chose, was to forego external tools and simply ask the OpenAI language model we are using to limit the types of weapons that characters can use. However, this results in a downside - we are artificially limiting the produced narrative based on the needs of the game's implementation.

Before commencing the design phase of our prototype, we dedicated some time to trying to figure out what ChatGPT's capabilities were (in relation to our purposes), and how we could effectively leverage its strengths. Initially, we were hoping to be able to use it to not only create a narrative from what is happening in the game, but also to keep track of the positions of the various enemies, obstacles, and the player. The idea was to maximise the amount of data that could be fed to ChatGPT as input, as the more it "knew", the more relevant a narrative it could create. Furthermore, if it could keep track of positions, we would be able to think of our game as a "thin client" [49] that merely rendered a complex game state which existed on our ChatGPT backend.

Our initial experiments with ChatGPT's capabilities convinced us that such an approach was not possible. Repeatedly, ChatGPT rendered versions of a 2D grid that went against our explicit instructions. For example, when instructing it to render a grid with a set number of obstacles, it would instead create a grid with wildly fluctuating numbers of obstacles - sometimes too many, sometimes too few, but almost never the quantity that we requested. Additionally, it had trouble keeping track of game piece positions, falsely believing that two creatures were beside each other when they weren't, or that the player could not attack a creature when, in actuality, it should have been possible due to proximity. Appendix A is an example

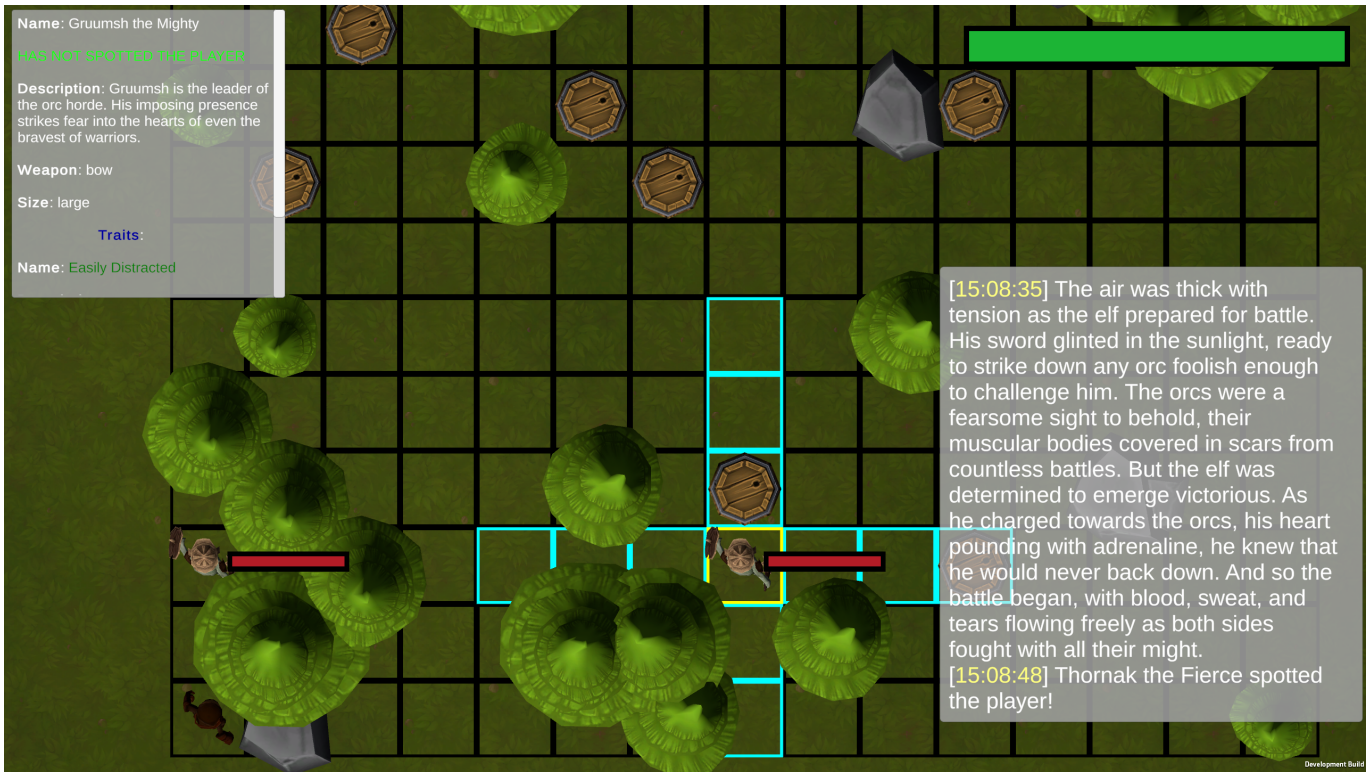


Figure 5.3: Screenshot of the prototype being played

of such a conversation with ChatGPT, demonstrating its shortcomings.

The discovery of these shortcomings set us on our path of attempting to sensibly leverage ChatGPT's ability to produce a narrative and offer some randomness to the instantiation of enemies, all while keeping a controlled game state within our Unity project. We needed to facilitate both the core gameplay's need for some variety and the need to integrate what happens during gameplay appropriately into the narrative. In order to accomplish this, we designed a system called "Traits".

5.3 Traits

We knew from early on in our work that we wanted our game to have some type of "status effects" system. Since every status effect could have a name describing its particular effect we decided that feeding them to ChatGPT would add some beneficial context and help it create appropriate narrative.

Over the course of our prototype's development, we conceived of and implemented a series of traits:

- **Burning**, caused by an exploding barrel, causes the creature to take damage every turn for as long as the effect is active
- **Knockdown**, can be triggered when a creature is attacked, forces the creature to forego their turn for as long as the effect is active
- **Drunk**, causes enemies to both deal and take more damage

- **Sober**, contrasting with **Drunk**, causes enemies to both deal and take less damage
- **Easily Distracted**, instills a chance that the enemy's turn will be skipped
- **Hungover**, if using a bow this trait instills a chance that the enemy's attack will miss
- **Short Sighted**, affects the range at which the enemy can possibly spot the player
- **One Armed**, if using a melee weapon this traits instills a chance that the enemy's attack will miss

At the beginning of each game, certain traits have a possibility of being assigned to enemies at random.

Traits are a necessity for our game for two reasons. Firstly, they add some much-needed variety to the game - without them, every occurrence of playing the game would be too similar. Secondly, since every trait's name is essentially an adjective, they can be injected into and interpreted by ChatGPT in order to enhance the narrative and appropriately use the currently applicable status effects in its creation. Appendix B details how we inject relevant traits into our generated prompt when an enemy is killed, in order to request of ChatGPT a narrative description of the kill that appropriately incorporates the currently-active traits.

Later on, our descriptions of traits for ChatGPT's purposes expanded. Sometimes, the mere adjective (name) of the trait was not enough for ChatGPT to grasp the full context of what had occurred in the game. For example, the killing of an enemy suffering from the "burning" trait would sometimes cause ChatGPT to concoct a piece of the narrative involving the protagonist suddenly having a magical, burning blade that would permit them to incinerate their foes. This necessitated a new property on our Trait class called `LLMDescription`, whose provision to ChatGPT is automatic (Appendix B), provided that it is present. Appendix C demonstrates how a series of `Traits`, existing as a member variable on a creature, are formatted for ChatGPT's consumption.

To illustrate the point, the "burning" trait's `Description` and `LLMDescription` values are as follows:

```
Description = "This orc is really hot";
LLMDescription = "This orc is on fire because, earlier, they were
↪  near an explosive barrel when it exploded";
```

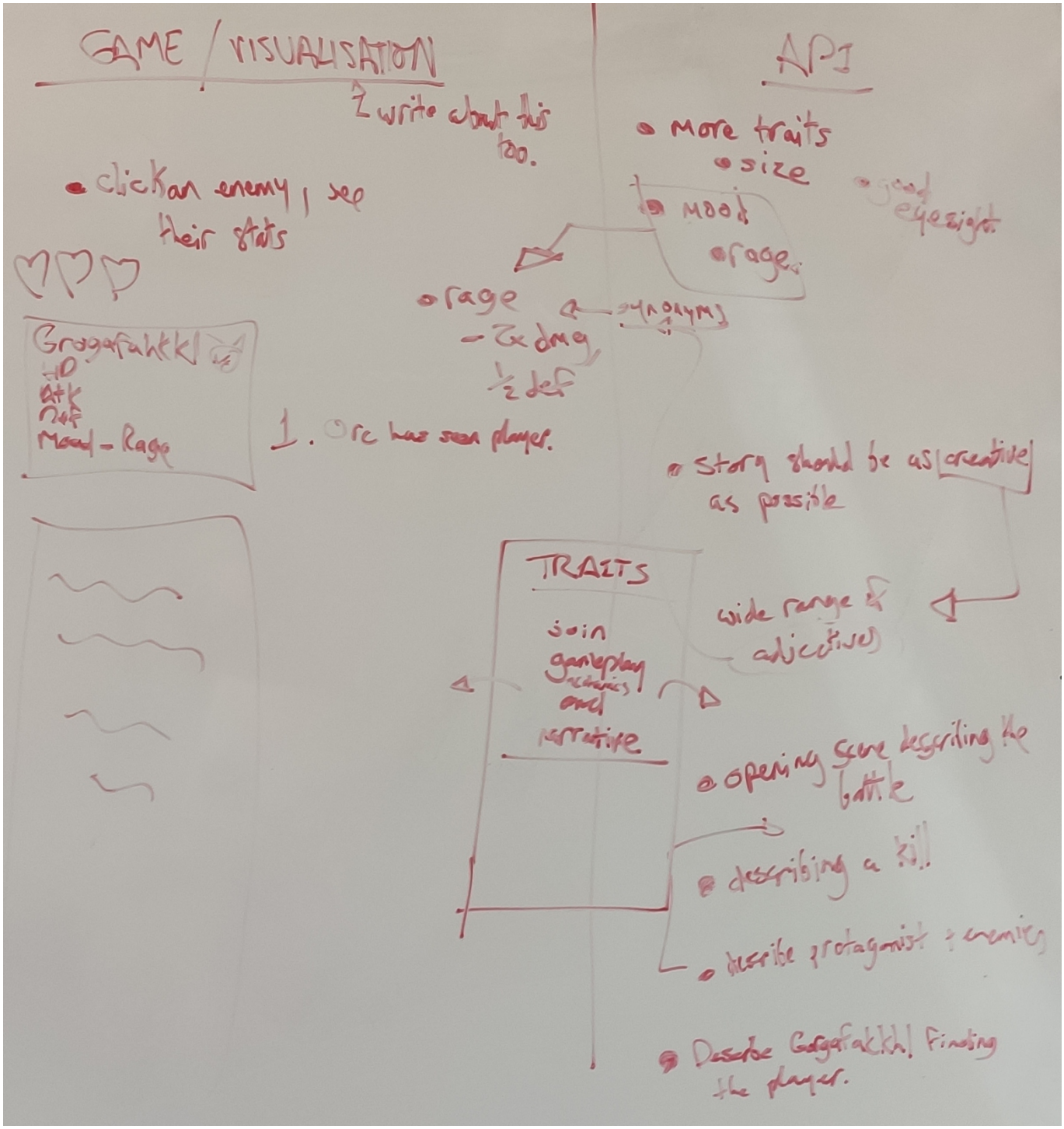


Figure 5.4: Early whiteboard session depicting our concept for traits

5.4 A turn-by-turn account of the game

Upon lauching the game, there is a slight delay as ChatGPT is accessed and the opening scene (the beginning of the story) is outputted for the player's perusal. The player always spawns in the bottom-left-most cell on the grid. Numerous obstacles are scattered throughout the map, including the aforementioned barrels which, upon being struck, explode after one turn.

The player's goal is to defeat all of the enemies, of which there are three or four. Obstacles obscure enemies' view - enemies will not be able to see the player through trees, rocks, or barrels. Enemies' default behaviour is to patrol - they have a starting position, and an end position. As long as they have not seen the player, they patrol indefinitely, going back and forth between their randomly-chosen goal destination and their starting position.

Enemies can only see a certain distance. This distance is determined by their traits - an enemy with the short sighted trait can see only one tile away, whereas an enemy without this trait can see two tiles away. Enemies "look" in the four cardinal directions only - these are the same directions that they can attack in. However, once they have spotted the player, they have a certain attack distance, determined by their weapon. Figure 5.3 demonstrates this - upon clicking an enemy, their attack range is signified by the tiles they can hit with their weapon becoming highlighted with a cyan outline. Enemies who have spotted the player always know where the player is from that point onwards, and will endeavour to chase and attack them. This continues perpetually until either the enemy or the player is defeated.

The player and enemies have limited movement and attack capabilities per turn. The player can move up to two times, and attack once, on their turn. Enemies can move only once, and attack once, on their turn. Both the player and enemies have a certain amount of "health" - being struck causes damage to this health. An entity is defeated when their health is completely depleted.

Upon defeating an enemy, the player is asked how they wish to dispatch their foe (Figure 5.5). This is an opportunity to be creative, with the goal of having ChatGPT effectively utilise the player's input to describe how the enemy is defeated. Furthermore, since we are informing ChatGPT about the traits that the dispatched enemy possesses, we are attempting to generate a narrative that appropriately interweaves these traits into the description of how the enemy is defeated (Figure 5.6).

The game continues until either all of the enemies, or the player, are defeated. The controls can be viewed in Appendix I.

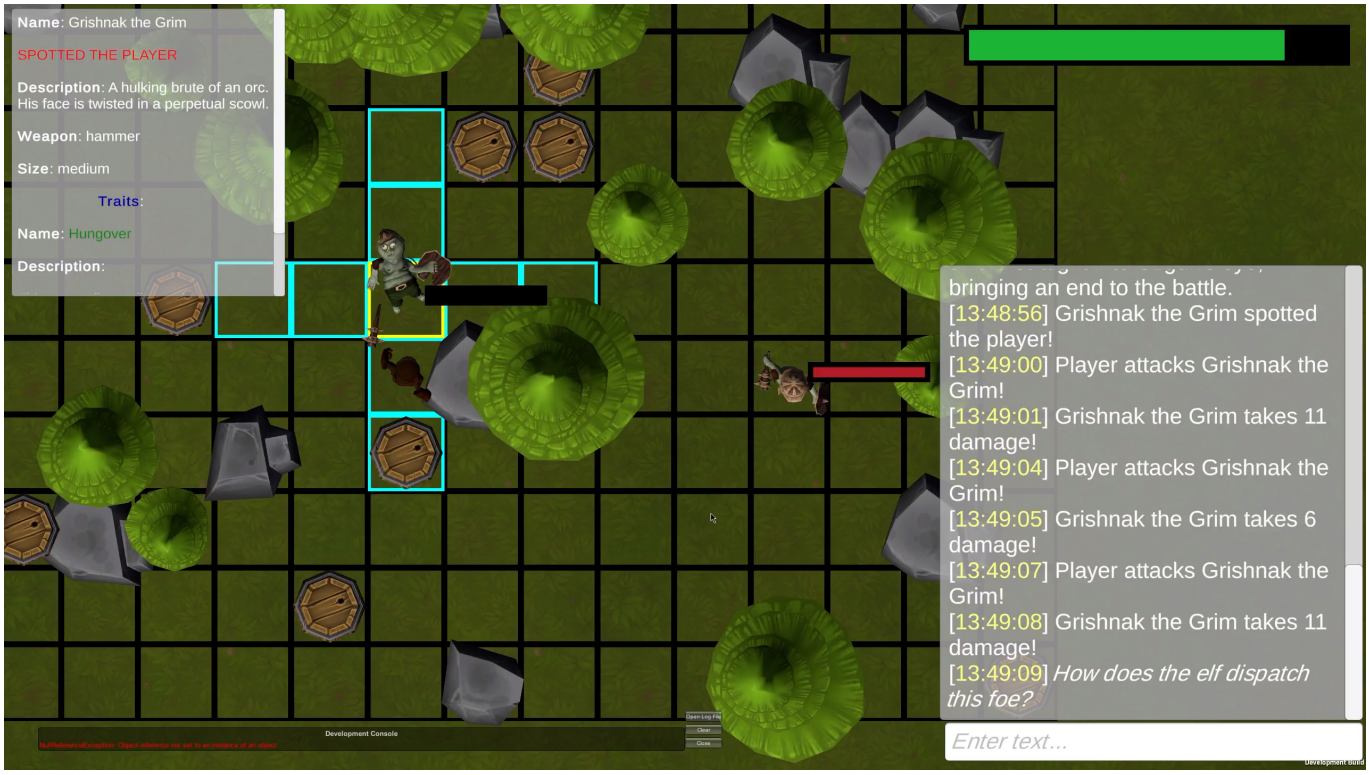


Figure 5.5: Example of the player being prompted to describe how they dispatch an enemy

Figure 5.6: Example of generated narrative upon dispatching an enemy

6

Evaluation

6.1 Approach to evaluation

The prototype that was developed throughout this project was a game prototype, and the project is focused on the potential uses of ChatGPT in video games. As such, the prototype had to be evaluated as a game. The aim of the evaluation was to learn how the addition of an AI-based dynamic narrative generation impacted the gameplay experience. This was achieved through a combination of two main aspects of our evaluation:

- Creating a contrast between a gameplay experience with an AI-generated narrative, and one without
- Collecting both qualitative and quantitative data about the experience from the playtesters' perspective

6.1.1 Contrasting gameplay experiences

One of the main challenges in the evaluation is to focus on the parts that we want to evaluate, reducing the amount of variables. The prototype developed was, as far as video games go, quite small. In the scope of this project it was impossible, and perhaps unnecessary, to create a complete, full game experience. In the evaluation, the aim is specifically to see how AI-based dynamic narrative generation impacts the game experience, and therefore our prototype had some basic gameplay systems and mechanics, but the focus is on narrative generation.

At first, we had considered using an existing game, such as Rimworld [50]. This game has game agents, so-called "pawns", that can be both player controlled, or non-player controlled, similar to enemies in our prototype. These "pawns" also feature a similar system of traits, and can interact with each other, which means that the game could have a similar implementation of AI-based narrative. The goal was to create a contrast between a game that could potentially have this implementation, but does not; and our prototype, which does, while also having similar gameplay systems.

However, using an existing, popular and polished game would introduce many additional variables to the evaluation, and could potentially skew the test results.

Since Rimworld is a very complex game, it is possible that the playtesters would focus on how vast the difference is in amount of content that the two games have, and this could have a negative effect on the evaluation results. It is also possible that, because of this, the playtesters would find it hard to focus on and evaluate the system that we want to test. As such, it was decided that we would not be testing our prototype with an existing game, and instead, we would *strip down* the prototype of and present the playtesters with two version of our prototype: one without the dynamic narrative generation system, with only the base game mechanics, such as movement and combat; and one version with the dynamic narrative generation system. The aim with this was to exclude unnecessary variables, and put the focus of the evaluation onto how the addition of this system impacts the gameplay in particular.

In attempting to decide the right approach with our evaluation, we considered how the developers of 1001 Nights [29] approached this problem. For 1001 Nights, playtesting was done via a physical showcase setup at exhibition sites in China. From this playtesting, data was gathered. The authors point out that the quantity of data gathered made it infeasible to evaluate the quality of the story content, so the authors decided to compare the the players' "achievements" (how successful they were at finishing the game with a favourable outcome) with the average number of inputs per play session, ending in either success or failure. By doing this they investigated the impact of engagement with the storytelling in the game on the "achievement level" attained. Three categories of "achievement level" were specified - making no progress, making some progress, or finishing the game successfully.

The authors found that, while there was some randomness at play due to the tools used, a correlation existed between player engagement and overall achievements. This was a favourable outcome, and one that we aimed to recreate in our own prototypes. However, we ultimately decided on a different approach to evaluation, since it was possible to completely disable ChatGPT integration for the sake of gauging its merits in our prototype. If the developers of 1001 Nights had disabled the functionality provided to them by gpt-2.5, their game would have ceased to function. As our prototype has a combat system, we opted to compare how the prototype without ChatGPT compares with the prototype with ChatGPT.

6.1.2 Quantifying the gameplay experiences

Another important challenge of conducting evaluation for this project lies in the inherent subjectivity of video game experiences. It is difficult to create a meaningful quantitative metric to evaluate this experience. While metrics such as the playtime and degree of game completion exist, as discussed previously, they might not be able to convey the full picture. Additionally, they lack descriptiveness as to why the results ended up being what they were. Qualitative methods, such as retellings, pose a different challenge. Players might be more inclined to reflect negatively on their experience, due to the incompleteness of the prototype, and that might not necessarily provide enough information to assess the success of AI-based storytelling. This approach requires a careful consideration of questions that the players will be

asked, and an analysis of their input that will help us only focus on evaluating the aspects of the game that interest us, and not necessarily the game experience as a whole. As a result, in order to conduct a meaningful playtesting evaluation, a combination of both qualitative and quantitative metrics is required.

As a solution to this challenge, a set of questions was developed with answer options being a list of options on a Likert scale. The test subjects were presented with statements, and they would select how much they agree with the statement with the numerical values from one to five, ranging from "Completely disagree" to "Completely agree" respectively. In addition to this, in order to collect qualitative data about they experience that could later be examined together with the quantitative data, they were presented with a few free-form answers, where they could describe their experiences while testing the prototype in more detail.

The questions presented to the testers could be roughly divided into three groups. The groups were the following:

1. Impact of ChatGPT's addition on the gameplay experience
2. Impact of ChatGPT's addition on the gameplay experience while replaying the game multiple times
3. The playtesters' experience of the narrative generated by ChatGPT

The first group included questions where the playtesters were asked to agree or disagree with whether ChatGPT's addition has had an impact on their gameplay experience, whether it was a positive impact and whether it was a hindrance, as well as a free-form question where they were asked for more detail on what exactly that impact was. The goal of these questions was to assess the overall changes that the addition of a dynamic narrative generation system had on the gameplay. The second group had a similar structure, but was tailored towards assessing whether this system had an impact on the *replayability* of the game in particular. The third group included questions regarding whether the playtesters felt like they had greater control over the narrative, whether they felt like it was naturally written, and how the addition of the dynamic narrative generation impacted the perceived uniqueness of the game.

In addition, the playtesters were asked which version of the prototype they preferred: the one with, or the one without ChatGPT and why, as well as whether they could see other games benefitting from a similar AI-based narrative system.

On top of these questions, the questionnaire included questions that would be used to correlate gameplay experiences of specific playtesters with their familiarity with a number of game genres, as well as their general gaming patterns, specifically a question regarding how often they play video games. This would enable us to assess whether the AI-based dynamic narrative generation system was more or less suited to a specific type of player, or specific style of gameplay.

6.2 Evaluation process

Based on the approach described previously, an evaluation process was developed, that would include supervised playtests, where the playtesters at first would be asked to play the prototype version without ChatGPT; then they would be asked to play the prototype version with ChatGPT; and afterwards they would be asked to fill in a questionnaire, consisting of previously mentioned questions.

During the playtest, the playtesters were only be given instructions regarding the controls of the game, how the game systems function (mechanics such as exploding barrels, or systems such as the traits and how they influence the behaviour of enemies). Additionally, the playtesters were informed of their goal in the game: defeat all the enemies on the map. Additionally, after reaching this goal, before proceeding to the next playtesting stage, the playtesters were asked specifically to replay the game multiple times. This includes both versions of the prototype. Otherwise, the playtesters were not asked or steered in any direction while playing the game, with the goal being that they experience the gameplay *as is*.

As for the testers themselves, nine people had volunteered to participate in the evaluation of this project. They were all familiar with video games, and played them regularly, at least weekly, as a hobby. They were all familiar with role-playing video games, and most of them were at least somewhat familiar with strategy video games, narrative-driven video-games, and pen-and-paper role-playing games. This test group composition results in an evaluation with a focus on a group of playtesters with the following characteristics:

1. The playtesters have a lot of experience with video games
2. The games most of them usually play are similar in nature to the prototype, having similar game mechanics and concepts
3. The games most of them usually play are usually paired with a narrative

7

Results

As mentioned previously, there were 9 volunteers in total participanting in our playtesting.

7.1 Participant responses

7.1.1 Participant A

Participant **A** was the first player to test our game prototype. They were very familiar with video games, playing them multiple times a day. They had a high familiarity with all video game genres presented in the questionnaire, but had less familiarity with for pen-and-paper role-playing games. They considered that addition of ChatGPT had a significant and a very positive impact on their gameplay experience. The participant mentioned that seeing their own actions be reflected in the responses produced by ChatGPT increased their engagement a lot. Specifically, during the playtests, they had ignited an exploding barrel next to an orc and seeing how after defeating the orc ChatGPT included a mention of that event in its response, they mentioned that they appreciate this kind of player control over the narrative through gameplay a lot. The participant also rated the natural feeling of ChatGPT responses highly. When talking about how similar games could benefit from a similar system, the participant mentioned Rimworld[50], and how a similar system can be implemented as a mod for that game. They also proposed that ChatGPT can be given less "creative control" over the game, meaning that the story itself can be written by the developers, and ChatGPT builds on top of this story based within its boundaries.

7.1.2 Participant B

Participant **B** plays video games 1-2 times a week. They are very familiar with strategy video-games and role-playing video games, but are only slightly familiar with narrative-driven video games and pen-and-paper role playing games. The participant said that ChatGPT's addition had a huge impact on the gameplay experience, and that this impact was very positive. They mentioned that the addition made the event descriptions "a lot more fun", and appreciated the fact that it was customised based on enemy traits and their input, as well as the fact that it was

dynamic and changed between each game. They said that the addition of ChatGPT had a very positive impact on replaying the game multiple times, and it made the narrative less repetitive between play sessions and more unique. The participant noted that the addition was a slight hindrance, however. Specifically, while they still preferred the prototype version with ChatGPT, they would also prefer the descriptions be generated by ChatGPT without their input. The participant said the language was naturally written, allowed for more player control, and other games would benefit from having similar implementations.

7.1.3 Participant C

Participant **C** plays video games once a day. They are very familiar with strategy video-games and role-playing video games, but less familiar with narrative-driven video games. They are only slightly familiar with pen-and-paper role-playing video games. The participant noted that the addition of ChatGPT had a great and very positive impact on the gameplay experience. They mentioned that the addition "made the experience more open to creativity". They said it also allowed for more player agency, less repetitive when replaying the game multiple times. The participant also particularly noted how it made them want to "explore its boundaries and put in ludicrous inputs just to see what it does", and that it also "new dimension to the narrative, which is fun". However, the participant agreed that the system was also a slight hindrance, since the buffering time was noticeable, yet according to them it does not detract from the gameplay experience, and encouraged to hide the loading form the player with other game elements or player activities. At the same time, they felt that the language of ChatGPT was naturally-written, and that other games could benefit from similar implementations.

7.1.4 Participant D

Participant **D** reported playing video games multiple times per day. They expressed strong familiarity with narrative-driven video games, role-playing video games, and pen-and-paper games, but neither agreed nor disagreed with the statement "I am familiar with strategy video games". They claimed that ChatGPT's integration offered a "good narrative addition", and that "a lot of more lore and situation description was added". However, while strongly agreed with the notion that ChatGPT's addition positively impacted the experience of replaying the game multiple times, they also stated that the typing itself started to get "a bit old" after a while. They strongly agreed with the notion that ChatGPT's integration could benefit other games, and stated that RPGs in particular could "feel like a more role-playing game rather than a game with just common rpg elements". However, they experienced some incongruity when attempting to "slice" an enemy "cleanly in half" - the response from ChatGPT described the orc's bow being sliced in half instead. Participant **D** experimented with changing the genre of the story by entering "call an orbital strike" when defeating a foe.

7.1.5 Participant E

Participant **E** expressed familiarity with all categories of games we asked about, with the exception of pen-and-paper role-playing games. They strongly agreed that ChatGPT's addition was a positive one, claiming that it "freshened it up and made me think about the role playing aspect more instead of the actions of the game". They strongly disagreed with the notion that ChatGPT's addition was a hindrance for the game, and strongly agreed that ChatGPT's addition allowed for more player control over the narrative and that the narrative generated by ChatGPT felt like it was naturally written. The playtesting session with Participant **E** provides a potential example of language ambiguity adversely affecting the response returned by the ChatGPT API - at one point in the narrative, Participant **E** stated that they wanted to defeat their foe with a "right hook", referring to a punch. ChatGPT returned a response that involved the player defeating the foe with a bladed weapon instead; this inconsistency could have been due to how the language model interpreted the word "hook".

7.1.6 Participant F

Participant **F** played video games multiple times per day and expressed high familiarity with strategy video games and role-playing video games. They level of familiarity they expressed with narrative-driven video games and pen-and-paper role-playing video games was comparitively low. They strongly agreed that ChatGPT's addition was a positive one, and stated that it added "more immersive and fun gameplay". They specifically mentioned that ChatGPT's addition allowed for some comedy to be injected into the game. However, they did not particularly agree nor disagree with the notion that ChatGPT's addition positively impaced the experience of replaying the game multiple times. They neither agreed nor disagreed that ChatGPT's addition was a hindrance for the game, but agreed that ChatGPT's addition allowed for more player control over the narrative, that the narrative was naturally written, and that other games could benefit from a similar implementation of ChatGPT. They stated that ChatGPT's addition forced the player to take some time to respond to the player prompt, disrupting the flow of the game. They also mentioned that the ongoing text log displayed a lot of text.

7.1.7 Participant G

Overall, Participant **G** expressed a high familiarity with the categories of games we asked about, with the exception of pen-and-paper role-playing games, to which they claimed only a small level of familiarity. They claimed that ChatGPT's integration made it more interesting to interact with the "NPCs". They strongly agreed that ChatGPT's integration was a positive one, and claimed that it was "interesting and fun to see what happened if you play around with different stories". Interestingly, despite claiming to be less familiar with pen-and-paper role-playing games than the other categories of games we asked about, they draw a comparison between the creativity permitted to players in pen-and-paper role-playing games and the ChatGPT-enabled prototype, claiming that the ChatGPT-enabled prototype

allowed for a similar type of creativity. They also noted that the second time they played the ChatGPT-enabled prototype, they played as a character with a different gender, making that playthrough more memorable than the previous one.

7.1.8 Participant H

Participant **H** reported that they were very familiar with every category of game we asked about - strategy video games, narrative-driven video games, role-playing video games, and pen-and-paper role-playing games. Participant **H** also strongly agreed that ChatGPT's addition changed the gameplay experience and that the change was a positive one. They believed that ChatGPT's addition "increased the variability of possible events", but also stated that after a few instances of dispatching a foe they noticed some repetitions of phrases in the narrative. They expressed excitement at the ability of inputting "unusual" text to see how ChatGPT could "cope" with it. Participant **H** also made an interesting suggestion that seemed to be inspired by the structure of the playtest itself - they stated that they would prefer it if the option to submit player inputs could be enabled or disabled at will. The reason they gave for this is that they sometimes aren't in the mood to be creative while playing, and that the option to toggle either player input or ChatGPT integration on and off would be a beneficial feature.

7.1.9 Participant I

Participant **I** was the only respondent who preferred the version of the prototype without ChatGPT. They had many positive views on ChatGPT's addition - it made multiple playthroughs less repetitive; it allowed them to have "somewhat of an impact on how the story was written"; and ChatGPT's addition was a positive one. Despite this, the fact remained that their favourite type of game was competitive games (they specifically named FPS games, card games, and MOBA games) and, while playing the prototypes, they did not really notice or pay attention to the lore or prompts. They therefore preferred the more streamlined and simple experience that did not attempt to enhance the game with the Chat-GPT narrative. Like Participant **D**, Participant **I** communicated that it might be possible to change the genre of the story from the fantasy-inspired one we presented to a "cyberpunk" one.

7.2 Higher-level results

Appendix K shows a breakdown of how the entirety of participants responded to the various questions. A majority expressed familiarity with every category of video game we enquired about, though this was not the case for pen-and-paper role-playing games, with a majority neither expressing that they "agree" or "strongly agree" that they are familiar with these types of games.

A majority of respondents strongly agreed that ChatGPT's addition changed the gameplay experience. All agreed that the change was positive, with six of the nine respondents strongly agreeing with the sentiment. All also agreed that ChatGPT's

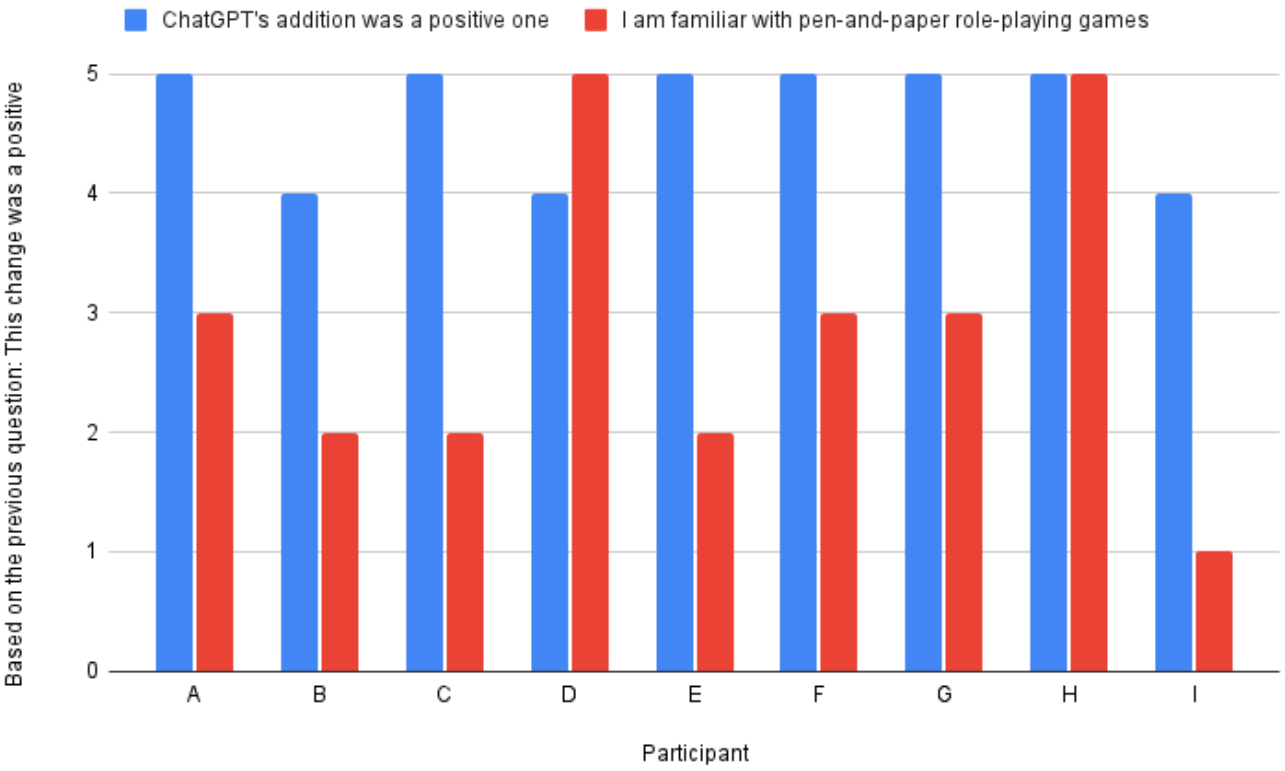


Figure 7.1: Pen-and-paper familiarity vs opinion on adding ChatGPT

addition made the game more unique. Though a minority agreed that the game was less repetitive when replaying the game multiple times, all but one participant claimed that ChatGPT's addition positively impacted the experience of replaying the game multiple times. Furthermore, a majority of players did not feel that ChatGPT's addition was a hindrance for the game, and all but one respondent claimed that ChatGPT's addition allowed for more player control over the narrative. Additionally, all but one respondent claimed that the narrative generated by ChatGPT felt like it was naturally written.

All respondents claimed that other games could benefit from a similar implementation of ChatGPT, and all but one respondent claimed to prefer the prototype version with ChatGPT to the one without it.

7.2.1 Correlations

We were interested to see if there would be a correlation between those who had a high level of familiarity with pen-and-paper role-playing games and those who viewed ChatGPT's integration as a positive one, as our approach was somewhat inspired by "Dungeons and Dragons" [40] - in particular, by players' ability to shape the narrative through verbal communication.

Our results (Figure 7.1) suggest a lack of correlation between familiarity with

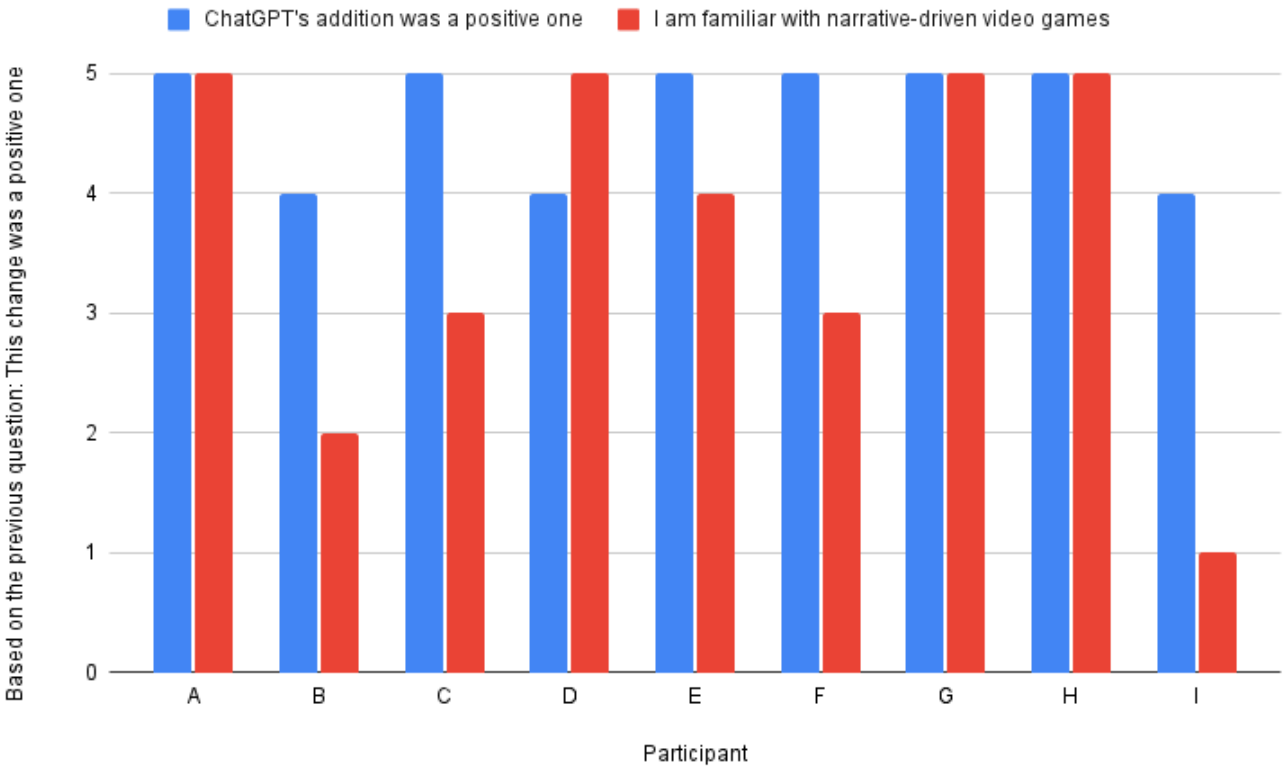


Figure 7.2: Narrative-driven video game familiarity vs opinion on adding ChatGPT

pen-and-paper role-playing games and the belief that ChatGPT's addition was a positive one.

There were similar findings when comparing respondents' views on ChatGPT's integration with familiarity with narrative-driven video games. The findings here (Figure 7.2) suggest a lack of correlation between being familiar with narrative-driven video games and believing that ChatGPT's addition was a positive one.

When comparing with role-playing video game familiarity, there may be a higher degree of correlation. Figure 7.3 shows a comparison with respondents' familiarity with role-playing video games. These results are much closer in nature, suggesting a possible close overlap between people who enjoy role-playing video games and those who would benefit from and enjoy the kind of ChatGPT integration we have in our prototype.

We did not notice a correlation between familiarity with narrative-driven video games or pen-and-paper role-playing games and the opinion that the narrative generated by ChatGPT felt like it was naturally written.

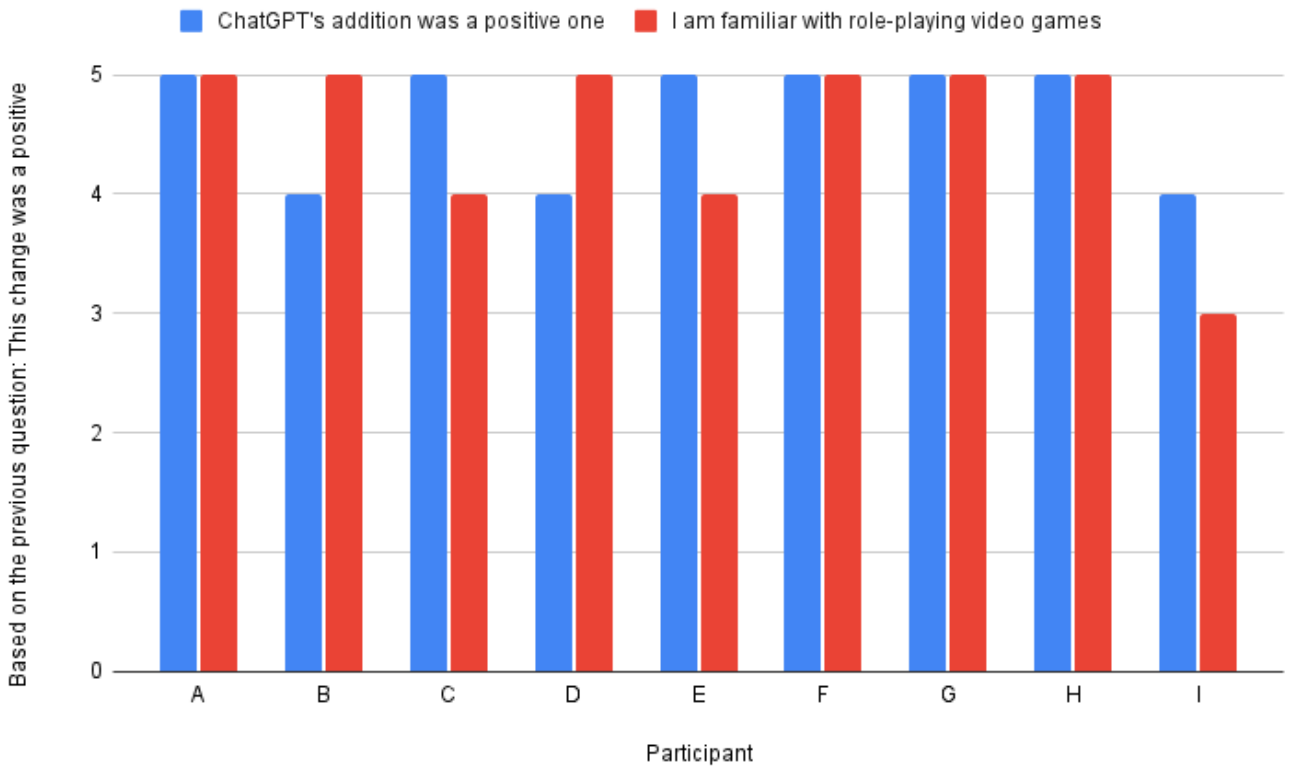


Figure 7.3: Role-playing video game familiarity vs opinion on adding ChatGPT

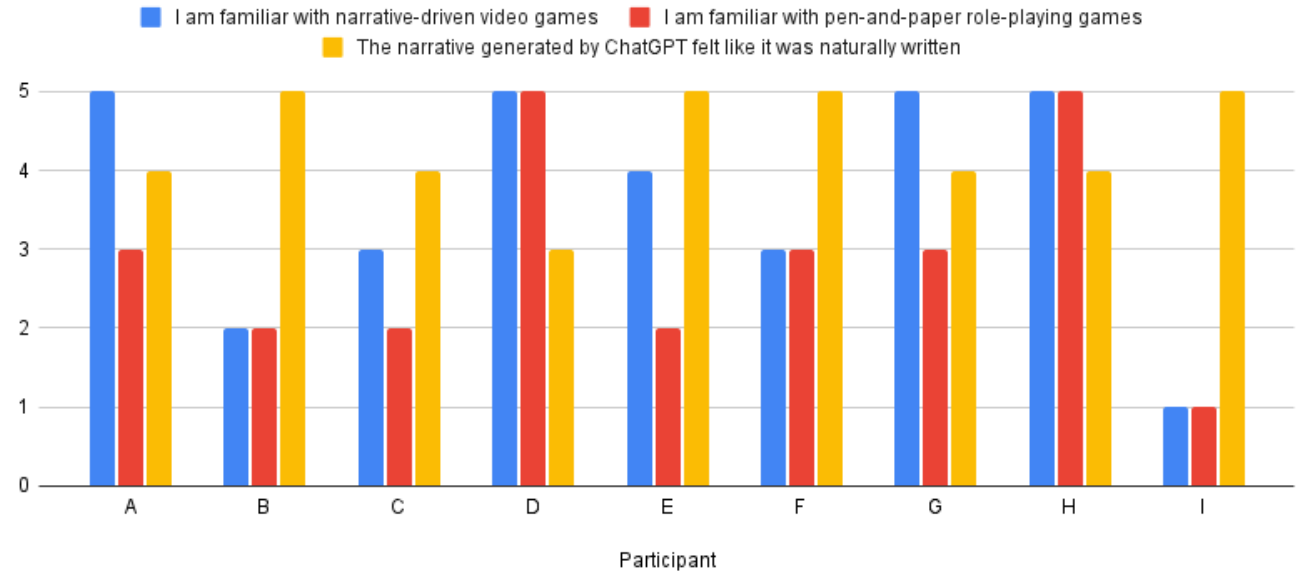


Figure 7.4: Whether the narrative felt naturally-written vs pen-and-paper and narrative-driven video game familiarity

7.3 Summary

Overall, the participant response suggests that ChatGPT's addition had an impact on the gameplay experience, and that it was a positive one. Correlations between more specific genre familiarity and responses of the participants are harder to find, however. Some correlation has been noticed among people who are familiar with role-playing games, yet due to a limited amount of participants, its hard to conclude firmly on this. However, one of the participants presented an interesting example. They had a significantly lesser familiarity with the game genres included in our questionnaire than other participants, and instead were mostly playing first-person shooter games and card games. Their response was significantly different from other playtesters, and they were the only participant to prefer the version without ChatGPT. The implications of this will be discussed in more detail in further chapters.

8

Discussion

8.1 Integrating ChatGPT

If one is considering integrating ChatGPT into a game, there are a number of considerations to take into account. This is at the heart of the first part of the problem we are trying to address - namely, "How easy or difficult is it to incorporate ChatGPT into our prototype?"

8.1.1 Requiring the API

The first concern, and perhaps the most obvious, is that connection to the API is necessary. Our version of the prototype references an API key that is stored within the source code in order to make calls to the API. For the purposes of our prototype, this was sufficient, as we were mainly concerned with a proof of concept and its overall viability than concerns regarding the securing of our API keys. Since a lot of research has already been done into this question (and industry standards established), we can summarise the options regarding the OpenAI API key as follows:

1. Continue to merely obfuscate the key and distribute it in the application (not secure)
2. Host the API key on a separate server and, rather than connecting directly to OpenAI servers client (game) side, connect to the interim server containing the keys and forward the request to the OpenAI API on behalf of the client
3. Require that users provide their own API key in an environment variable or plaintext file in a location readable by the game

To anyone with sufficient knowledge and patience, obtaining our API key from our distributed game would be trivial. For this reason, our current approach would not be viable for other game development teams and our current approach should be looked at as a proof of concept only.

Furthermore, as was mentioned in Section 5.2, there is the need to implement an actual REST client. As ChatGPT matures, the further development of more seamless libraries to interface with its APIs is a distinct possibility (as is the further development of more generic REST clients). While these things may ease the burden

of development, some effort to properly interface with the API in the game code will still be necessary.

That being said, we do not foresee there being substantial issues for a developer to be able to make the REST calls themselves to OpenAI's API. Unity's own UnityWebRequest [47] made it possible to create our own OpenAI API client from scratch, and the usage of mock data (saved from a previous instance of using the API) allowed for the API client to be developed in parallel with the game.

8.1.2 Input Sanitisation

Input sanitisation is another broad issue. It is not only a consideration for allowing users to input prompts of their own creation with ChatGPT; it is a potential point of concern for every application permitting user input of any kind.

ChatGPT's popularity is a relatively recent phenomenon, requiring its own security-related research to discover the dangers unique to it and large language model (LLM) usage in general. Adversaries can potentially manipulate prompts for nefarious purposes [51], and doing so would be even more straightforward in an application like ours where we are directly using user input as part of the prompts sent to the API. It may take some time before practises and tooling catch up with the current capabilities that are now available for general usage through these APIs.

8.1.3 Violating OpenAI's policies

Appendix B exemplifies our attempts at ensuring that we do not trigger any policy violations when using the API. However, there is always the possibility of this happening due to player input, as the screenshot of one of our playtesting sessions in Figure 8.1 demonstrates.

This highlights the possibility of ethical concerns, as well as how immersion can be broken by having the API report such policy violations. With time, the language models available for use may get better at responding to inappropriate requests in a way that does not violate any policies and would not break immersion for our purposes. In the meantime, if such issues want to be avoided, careful scrutiny is required of player inputs. However, even if this is done, it may not be enough to totally avoid the issue.

8.1.4 Self-management of context

The context of an ongoing conversation with the ChatGPT needs to be managed client-side. When ChatGPT is used via its web interface, the experience is seamless as the full context of the conversation is automatically utilised on the user's behalf. This is not the case when using ChatGPT via an API - every time a follow-up prompt is sent to the API, all of the context required to respond correctly to the prompt must also be sent.

This necessitates a method of keeping track of all of the required context over the

Figure 8.1: Violation of OpenAI Policy

course of the application's use. This context essentially takes the form of a running conversation log between the user and the application. In order to accomplish this we made use of a `Message` class (Appendix G), capable of containing both the `message`'s content and the `role`. The `role` property is how user-generated text is distinguished from ChatGPT-generated text - user-generated text has the "user" role, and ChatGPT-generated text has the "assistant" role.

Our approach is as follows: whenever a new prompt is sent to the API, the entire conversation history until that point is sent to the API. We iterate over every previous message, accumulating a series of `ChatGPTPosts` (Appendix H). This is all done in our `Post` method, as is the appending of the new prompt that we want a response to, and the commencement of sending a POST to the API (Appendix F).

Such an approach is somewhat crude. Depending on one's usage of the API, it may not be a good idea to send the absolute entirety of the conversation's history to the API due to the inherent token limit.

8.1.5 Token limit

While we didn't have any issues with the token limit, it is highly possible that a game of sufficient length and complexity would run into this issue. The model we used, gpt-3.5-turbo, supports a maximum of 4096 tokens. If we were to approach this limit, we would need to adopt a more selective approach to our management of context.

For example, we currently send any additional `LLMDescription`s of a trait every time an enemy possessing that trait is defeated in the game. A more selective approach would be to only describe the traits once initially, and then refer to them only by name later. This feature was implemented in our `Traits` class, but is not currently utilised (Appendix C).

Another consideration is the upcoming addition of support for up to 32,768 tokens to gpt-4. While this still means that there is a limit to how many tokens can be passed to the API and processed, it does at least expand how much developers would be able to do before they have to concern themselves with reducing the token usage of their game. However, there is always the possibility of a game being made with sufficient complexity and length to warrant the kinds token-saving techniques that have been mentioned here.

Another way of saving tokens would be to experiment with removing certain sections of the conversation context and see if they have any adverse, negative effects on the generated narrative. The entire JSON object, as described in Section 5.2, is part of this conversation history. While it is necessary to know and remember certain things about the enemies, perhaps the JSON object could be trimmed somewhat in order to save tokens.

Ultimately, this concern comes down to a tradeoff. Time and effort are required in order to experiment and ensure that negative consequences of trimming tokens in context are avoided. How this could be done, and how beneficial it would be, would come down to the particular game and its usage of ChatGPT.

8.1.6 Network requests

Our prototype routinely needs to communicate with the API. This occurs both at the very beginning of the game in order to populate the enemy objects with initialising data (mentioned in Section 5.2) and to generate narrative when an enemy is defeated. In both of these cases, a spinner is displayed to the user (Figure 8.2), signifying that player input has been momentarily paused.

Loading screens are a necessary evil in video games. They can exist for many reasons, not just network requests. For example, in some games, assets are routinely loaded into memory when the player moves from one area to another, requiring a momentary pause. Their drawback is that they break immersion by causing the player to regain their sense of self-awareness [52]. For this reason, game developers have increasingly devised methods of disguising their loading screens in an attempt to make them unnoticeable during gameplay; an example of this can be found in the game "Dungeon Siege" [52].

Such loading-screen-hiding solutions are often used to disguise the time needed to perform some kind of computation or loading into memory locally, rather than perform a network request, which is typically much slower than loading new content locally. Hiding a network request can be more challenging for the simple reason that a network request might take a good deal more time than performing a local load

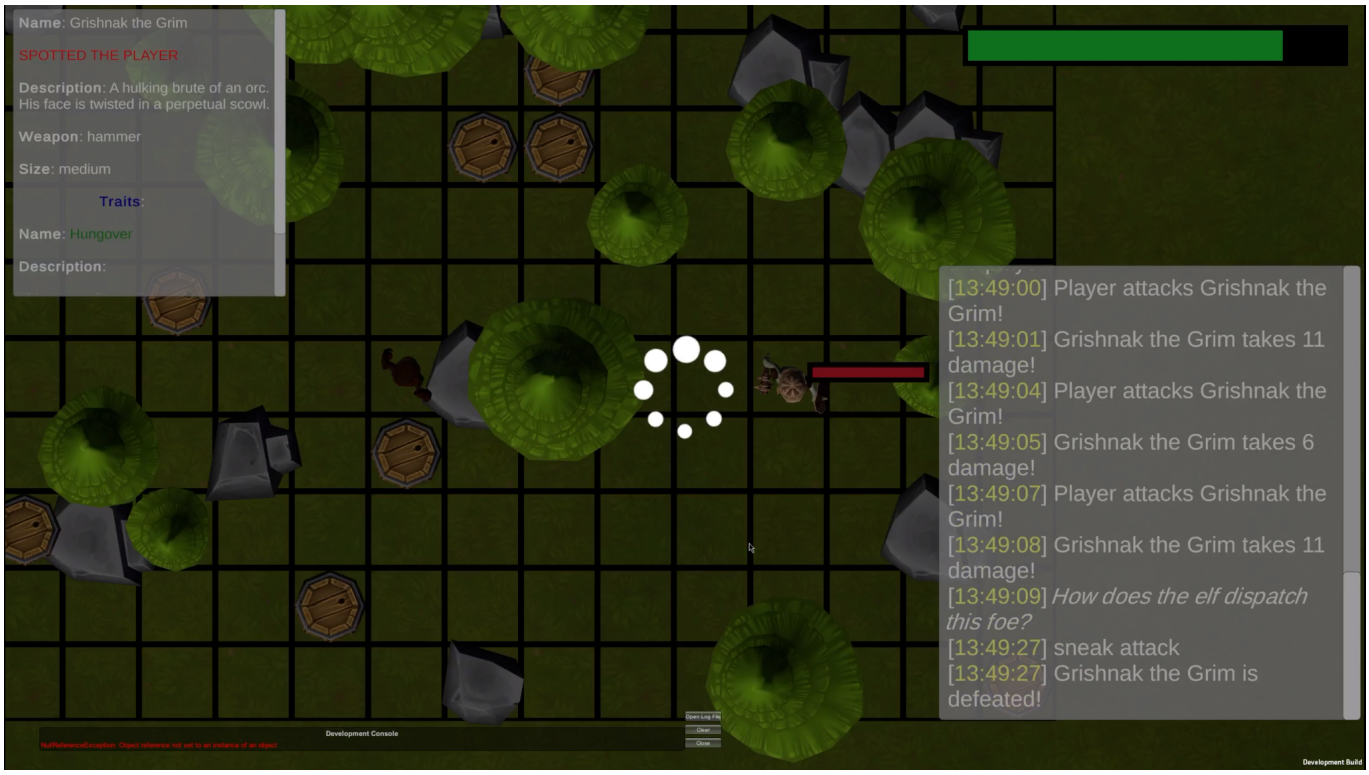


Figure 8.2: Prototype spinner during API network requests

into memory. In order to remove the loading time of narrative when an enemy is defeated, a creative solution may be required.

One potential solution is to simply pre-cache data from the API endpoint so that no network request is necessary during gameplay. This approach has an obvious drawback - facilitating player input will be much more difficult if the only possible responses are pre-cached. A possible middle-ground would be to do both - respond with a pre-cached response when one already exists, and perform a network request only when this is not the case. This would allow the pre-caching of common examples of player input, potentially removing network requests altogether in the majority of enemy dispatches. Additionally, certain other API requests could be pre-cached without such drastic consequences on the playback of the game - for example, the initial request populating the JSON object for game setup. A large number of cached game-setup scenarios could be generated before the game is played, allowing one to be chosen upon game launch, thereby removing the need for a network request at the beginning of the game.

8.1.7 Prompt engineering

As the prototyping process in Chapter 5 describes it, most of the game systems have to be kept track of by the game code, rather than delegated to ChatGPT, due to its inherent lack of game state tracking capabilities. As the amount of game elements that we had to operate with in the game grew, we had to inform ChatGPT of more and more things. This resulted in us having to take more and more care of how

we write our prompts to ChatGPT, in order to make sure it consistently generates responses in a correct format, and a correct context. This was important all the way from the beginning, with our first prompt. As the first request response that the game expects is a JSON object, we had to make sure that it only outputs JSON. Having it respond with something else would result in unexpected and unwanted behaviour of the code. After some tries, we ended up with a prompt that reliably generates JSON. However, as game developers we had to take this into consideration when developing our systems. It should also be noted that this would also potentially make it difficult to switch between models and model versions. ChatGPT's model, from a developer's perspective, is a "black box", and changing the version might break the prompts that were previously working correctly.

While partially an issue of input sanitisation, exposing ChatGPT directly to the players might also have unwanted consequences. As described previously, during the evaluation we had issues with the players triggering OpenAI's policy warnings. While an ethical issue in itself, such policy warnings and unexpected "masks off" moments with ChatGPT also create a great deal of immersion breaking. In order to minimise this, we had to tweak our prompts numerous times. Testing and rewriting prompts that inform ChatGPT of players' input and other game events is something that game developers will have to do in order to implement a similar system.

8.1.8 Genre suitability

We benefited from the genre of game that we created for our prototype - in particular, that it was a turn-based game. This meant that whenever we wanted to communicate with ChatGPT, all we had to do was display a spinner and force the player to wait until the response came back. In other types of games, particularly games that are not turn-based, this could potentially be far trickier to implement.

For example, if a developer wished to add this functionality to an first-person shooter (FPS), in which the player always has control over their character, suddenly halting all gameplay in order to wait for a network request would probably be considered unacceptable by a majority of players. This highlights a certain degree of "genre suitability" - some games may be better suited to ChatGPT integration than others. However, it is also possible that, in the case of an FPS, there are fewer ways that the player can influence the outcome of events (particularly if it was a game in which the player cannot input any text for use by ChatGPT). If that is the case, than perhaps some inspiration could be drawn from the types of pre-caching solutions mentioned in Section 8.1.6 - while gameplay is taking place, the generated narrative to multiple possible outcomes could be fetched and cached, ready for use as soon as the appropriate trigger requires that the narrative be rendered. This kind of tradeoff would need to be seriously considered for any game, turn-based or not, but could potentially be an even greater concern for those games with real-time player control.

8.2 Game Design constraints

8.2.1 Gameplay flow and immersion

One aspect of game design is the design of an immersive gameplay experience. This means that the interaction between the player and the game is seamless, and the player is engaged in the game, with the goal of making the player distance themselves from the real world and feel like they are a part of the game world instead. As Taylor [53] puts it, video games allow the player to "become deeply involved in the game as an experiential space". This requires that all elements of the game create a consistent *diegetic* experience.

The issues discussed in the previous section of this chapter, including network requests and genre suitability, create additional constraints for implementing a system of AI-based dynamic narrative generation, such as the one developed in this project. Disrupting player actions by having non-diegetic elements has a great effect on the player being able to be immersed in the game, as being deeply involved in a diegetic experience means that every non-diegetic element runs the risk of this involvement and the deep connection with the game that it creates to be broken. As has already been discussed, loading screens are a common culprit of this which the developers try to get rid of with various means, and as hardware used to run video games becomes more powerful, it becomes possible to remove them not only with clever design tricks, but also with plain computing power of modern computers. Older games, such as The Elder Scrolls V: Skyrim [54], have mods that allow the player to remove loading screens in the game world [55], to allow for an uninterrupted gameplay experience.

However, loading screens are not the only game elements that can influence immersion in a negative way. Any non-diegetic aspect of the game can do that, including game mechanics and user interface. In our game prototype, whenever an enemy was defeated by the player, a text window would appear, awaiting for the player to enter their prompt, before they can continue to play the game. This was noticed by playtest participants **D** and **F**. Participant **D** mentioned that having to repeatedly enter prompts "got a bit old", and the participant **F** directly said that it made them feel like the flow of the game was broken by this. A pop-up text field appearing like this instantly reminds the players that this is exactly what they are doing - they are playing a game. The player character stops moving around, they can no longer attack or do any actions, and the players are forced to come up with a description of what their fictional character would do with the fictional orc they just defeated. They have to actively distance themselves from the game world and the player character and come up with an answer to continue playing. This has a huge risk to break the game immersion. This issue is an example of a wicked problem: *How do we make user interact with the game system while reminding them as little as possible about the fact that it is not real, and they are just playing a game?*

Solving this issue is out of scope for this project, and requires extensive research of its own. However, some solutions might be worth exploring if a researcher ever

wants to conduct this kind of research. One of the participants of our evaluation, participant **B**, mentioned that they would rather see ChatGPT generate descriptions automatically, without their input. This solution might be more or less suitable based on game genre, and based on what kind of interaction with the player the designer wants to create, but in a more dynamic game, one where breaking the flow of the game might have even bigger consequences, such as a first-person shooter game, having ChatGPT generate description in the back might be a better option. Instead of player input, the game might generate events based on the weapons the player is holding and the environment they are in, how accurate their aim is, or any other metrics that are present in the game. Similar approaches can be adopted in other fast-paced game genres, such as racing games.

Alternatively, some modern trends in video games can provide the designers with more ways to indirectly, or at least less directly, collect input from the player in order to customise the description generated by ChatGPT. Augmented reality and virtual reality technologies provide a plethora of sensors and trackers, such as movement and eye motion trackers. Players can not only use these trackers as game controllers, but data from these sensors can also be used for the dynamic narrative generation system to communicate with the players. For instance, gestures and limb movements can be used by the players to give signals to a backend AI generating the narrative. In a similar way, instead of collecting the prompts from the players through typing text, such as it was implemented in our project, voice controls present great potential to achieve the same goal, while also potentially minimising the negative impact on player immersion.

8.2.2 Genre constraints

Out of the nine participants that we had in our playtests, eight of them said they preferred the version with ChatGPT. However, one of the participants, participant **I** still said they would prefer the prototype without ChatGPT. And this presents us with an interesting insight into what type of gamer, and what type of game is suited for a dynamic narrative implementation. The game prototype developed as a part of this study was deliberately chosen to be a turn-based strategy / role-playing game prototype. It's this type of game that usually puts a bigger emphasis on the story, the narrative, and how the player interacts with and influences it. This is also supported by the data from our evaluations as visualised in Figure 7.3: we can see that players with a familiarity in role-playing games felt that ChatGPT's additions were a very positive change to the gameplay experience.

Pinchbeck et al. [56] have conducted a study of Half-Life 2 players' behavioural patterns. They conclude by saying that in a first-person shooter game, narrative devices and structures are left in the background, with game agents being in the spotlight of player attention instead. In our evaluation, this view was supported by participant **I**, who was, as they told us, mostly playing first-person shooter, and action games. During the playtest, they would consistently pay significantly less attention to the narrative elements of the game, including the responses that ChatGPT was generating, than other players. While this is only one participant of

the evaluation, this still might give us an insight into genre constraints and genre suitability of a dynamic narrative generation. Perhaps, if the bulk of the attention of the players is elsewhere, and with the gameplay immersion constraints described in the previous section, it is not worth it to attempt an implementation of a similar system in a first-person shooter game, or a genre where similar player behaviour patters are present.

8.3 Benefiting from ChatGPT

Section 7.2 details some high-level results from our evaluation. From this kind of high-level overview of the results, certain insights can be gained.

Since there was a broad mix of familiarity-levels with pen-and-paper role-playing games, the fact that respondents generally reported that ChatGPT's addition was a positive one suggests that it is not just pen-and-paper role-playing game player who see the value in ChatGPT's addition to our prototype. Interestingly, the same can be said for familiarity with narrative-driven video games.

There are two possibile insights to be gained from this. Neither possibility nullifies the other. Firstly, it is possible that some respondents, despite a lack of familiarity with narrative-driven video games or pen-and-paper role-playing games would actually enjoy them if they started to become more familiar with them. Secondly, it is possible that ChatGPT's addition offers benefits to those who don't (or wouldn't) enjoy pen-and-paper role-playing games / narrative-driven video games in the first place. This second possibility is interesting because it means that ChatGPT's appeal could (for some people) be one that exists separately from other forms of storytelling in games. It is possible that this kind of narrative generation would provide benefits to players who normally would not care about any form of narrative at all.

There seemed to be a much stronger correlation between finding ChatGPT's addition to be a positive one and familiarity with role-playing video games. However, it must be noted that nearly all respondents expressed a degree of familiarity with role-playing video games. In order to properly test this correlation, it would be necessary to conduct more playtesting sessions, and hear the opinions of people who are not familiar with role-playing video games, to see if this patterns persists. Still, it is possible that those with an interest in role-playing video games are particularly attuned to the narrative benefits that ChatGPT can offer.

With respect to whether the narrative felt like it was naturally written, there seemed to be no correlation between this belief and familiarity with narrative-driven video games or pen-and-paper role-playing games. This suggests that ChatGPT may have a broad appeal and, if applied potently, could provide value to a wide variety of players.

8.4 Summary

When adding ChatGPT to our prototype, there were numerous considerations - some (requiring the API, sanitising inputs, managing context, limiting token use) have been purely technical. Some are purely design-oriented (genre suitability, gameplay flow, immersion). Interestingly, some straddle both concerns (delay with network requests, the violation of OpenAI policies). This is important to point out - the ease or difficulty with which ChatGPT can be incorporated into a video game is not merely technical or merely concerned with design - it affects both aspects.

When playtesting and gauging ChatGPT's benefits, the results highlight two possibilities, which could co-exist - the first is that players who enjoy role-playing video games in particular have a lot to gain from ChatGPT integration. The second is that all players, regardless of whether they typically enjoy narratives in games, have a lot to gain from ChatGPT integration. Further playtesting, with more respondents, and possibly a greater variety of prototypes, would be required in order to further delve into this topic.

9

Conclusion

9.1 What are the constraints and affordances of incorporating ChatGPT?

In answering this question, there are a number of considerations. On one hand, there is definite work required to implement ChatGPT in a game for this purpose. In Chapter 8 we addressed many of our concerns in integrating ChatGPT. Concerns regarding interfacing with the API, securing access tokens, sanitising inputs, and self-managing context, all while trying to ensure that the integration is a good fit in the first place, will no doubt be time-consuming and potentially costly. ChatGPT also present a somewhat novel problem in game development: prompt engineering. Game developers will have to work with ChatGPT's interpretations of information that is provided to it, and make sure that it responds in a way that is suitable for their code. This also includes potentially making sure that ChatGPT "plays along" with whatever role it is designed to have. This becomes significantly more difficult if the game designers want the players to influence ChatGPT directly. However, we stress that as a two-person team, we found it perfectly doable to create a prototype that, in playtesting, had noticeable benefits for playtesters. It is for this reason that we find integrating ChatGPT not overly-difficult and would recommend that developers interested in adding it to games to do so.

However, we have mentioned design considerations as well - genre suitability and the potential breaking of immersion are definite concerns. It is not guaranteed to be worthwhile - one should consider the type of game they are making as an important factor in this decision. As we mentioned in 2.3, one could adopt the approach where they allow ChatGPT's capabilities be an actual factor in the design of the game itself. This would follow the advice in [12], and improve the possibility that ChatGPT's capabilities are utilised beneficially. Playtesting shows its value here - since adding ChatGPT to games is a recent phenomenon, playtesting to gauge the effectiveness and benefits of its integration is hugely beneficial.

It should also be noted that integrating ChatGPT like we have done could raise the costs of development, which should be taken into consideration by the game designer. This comes down to the circumstances in which the game is developed. In our case, we compared a version of our prototype with no narrative to a version

that had a narrative purely thanks to ChatGPT. However, there is an alternative comparison to be made between a version of a game with a narrative written by a dedicated writing staff and a narrative written by ChatGPT. We propose that ChatGPT can be of particular benefit to small development teams for this reason - [25] gives the extreme example of Star Wars: The Old Republic having a large writing team for a number of years before the release of the game. This is out of reach for many developers; in this type of situation, ChatGPT's integration could drastically lower the cost of development, though it is possible that the quality of writing would suffer as a result.

9.2 Does incorporating ChatGPT into to such a project have a positive influence on the player experience?

In answering this question, we must refer to our playtesting sessions and results. The results of our playtesting were positive. The overwhelming majority of playtesters rated the addition of ChatGPT as a very positive impact on the game. While the small participant pool makes it hard to draw strong conclusions, it can definitely be argued that an implementation of a simple AI-based dynamic narrative generation system is beneficial to a game experience. In comparison with the first prototype, that had no narrative save for a basic, unchanging introductory story paragraph, the playtesting suggests that ChatGPT can result in a much more unique, engaging narrative that allows for a much greater feeling of player control.

Ultimately, ChatGPT's integration into our prototype was worthwhile. It benefited the game and was received very positively by playtesters.

9.3 Future work

Our playtesting suggests that there might be genres that are more or less suited for this system, though more research might be needed to evaluate this. We suggest a more thorough evaluation with players of, and in context of, more varied array of game genres.

Since our prototyping focused on compating ChatGPT's integration with the outright absence of narrative, another beneficial comparison would be that between narrative written by actual humans and narrative generated by ChatGPT. This would be interesting to look into for the aforementioned reasons of cost and the possible scope and ambition of a project, particularly when worked on by a small development team.

As we wrote of in previous sections, there are concerns regarding gameplay flow and immersion. Possible work here could include calculating how damaging to immersion having these kinds of network requests are, which games suffer from this the most, and which kind of techniques can be utilised in order to mitigate these

problems.

Further playtesting could be done to explore the possible correlation between people enjoying ChatGPT's addition in games and familiarity with role-playing video games. It is possible that ChatGPT offers unique benefits to these types of video games, which would make sense given the focus on narrative they often have.

There is also a potential of a research into how prompts should be designed for video games that use ChatGPT or similar models. While the researchers would potentially have to deal with the specifics of each model, developing a set of prompt-related guidelines for game developers to follow if they wish to utilise AI-based narrative generation could provide an important assistance tool. This could result in more games including these narrative systems, and would create a potentially compelling solution for smaller game development studios.

Additionally, it would be interesting to see if similar prototypes could be made without any required network access at all. If an alternative to ChatGPT could be run entirely locally on the player's machine, it could mitigate many of the issues that we came across during our development and playtesting, such as those related to network latency, and perhaps even those related to OpenAI policy violations.

Finally, we believe there is room for either an expanded version of our prototype, or a new prototype with more context for ChatGPT to utilise. More traits that can be used in descriptions, a greater variety of enemies, more varied environments, and more instances of ChatGPT generating narrative are potentially hugely beneficial. We concerned ourselves largely with how enemies are defeated, but could have gone much further. For example, narrative passages could have described the moment in which an enemy spots the player and begins chasing them; they also could have been used to describe the player defeating an enemy who hasn't found them yet, implying that the player utilised stealth and cunning rather than raw, brutish force. Since ChatGPT appears to appropriately utilise the context we provide to it a great deal of the time, additional context could provide great benefits for video game narratives.

9.4 Final words

In summary, as Artificial Intelligence, and especially language models, are currently gaining both public and academic momentum faster than ever before, we would expect to see more and more Artificial Intelligence applications in video game narratives. There is still a great amount of research that can be done in this area. AI-based dynamic narrative generation in a wider variety of genres, perhaps even genres that are traditionally less associated with a complex narrative, is one possible research topic. Additionally, reducing technological and game design constraints of a network-based AI implementation, or possibly even implementing a more localised solution within the constraints of typical consumer hardware, is another possibility.

As a part of this thesis, a ChatGPT-based AI-based dynamic narrative genera-

tion system was developed in a simple game prototype. Based on our evaluation, we argue that it has the potential to have a positive impact on the game experience from players' perspective. Our evaluation suggests that this positive impact is significant. However, due to the limited number of participants, a more wide-scale evaluation of a similar implementation might be required to draw more firm conclusions. In this thesis, we have also discussed the various constraints, both from a technological and game design perspective, that the developers might encounter. As such, we argue that AI-based dynamic narrative generation systems can be a useful tool for game developers, especially smaller developers with potentially less resources for a fully fledged complex narrative writing team. However, before developers commit to this tool, they should carefully consider whether their particular game, game mechanics, and target audience will appreciate this, since some game genres and gameplay styles might be more or less suited for this system. We argue that when it complements the game context, and is planned and executed properly, an AI-based dynamic narrative generation system (based on ChatGPT or potentially a different tool) can provide benefits that exceed its associated inherent challenges and disadvantages, and can have a positive impact on players' experience in the game.

Bibliography

[1] OpenAI. "ChatGPT." Accessed on: Feb. 14, 2023. (Feb. 2023), [Online]. Available: https://chat.openai.com/.

[2] Latitude. "AI Dungeon." (), [Online]. Available: https://play.aidungeon.io/main/home.

[3] M. P. Eladhari, A. Sullivan, G. Smith, and J. McCoy, "Ai-based game design: Enabling new playable experiences," *UC Santa Cruz Baskin School of Engineering, Santa Cruz, CA*, 2011. [Online]. Available: https://www.researchgate.net/profile/Anne-Sullivan/publication/267295551_AI-Based_Game_Design_Enabling_New_Playable_Experiences/links/5453b0ea0cf2cf51647c20b3/AI-Based-Game-Design-Enabling-New-Playable-Experiences.pdf.

[4] OpenAI. (2015), [Online]. Available: https://openai.com/about/.

[5] N. Carlini, F. Tramer, E. Wallace, *et al.*, "Extracting training data from large language models.," in *USENIX Security Symposium*, vol. 6, 2021, p. 2633.

[6] D. Brand and P. Zafiropulo, "On communicating finite-state machines," *Journal of the ACM (JACM)*, vol. 30, no. 2, pp. 323–342, 1983.

[7] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[8] J. Freiknecht and W. Effelsberg, "A survey on the procedural generation of virtual worlds," *Multimodal Technologies and Interaction*, vol. 1, no. 4, p. 27, 2017. [Online]. Available: https://www.mdpi.com/2414-4088/1/4/27.

[9] B. Bontchev, "Modern trends in the automatic generation of content for video games," *Serdica Journal of Computing*, vol. 10, no. 2, pp. 133–166, 2016.

[10] C. Aeronautiques, A. Howe, C. Knoblock, *et al.*, "Pddl| the planning domain definition language," *Technical Report, Tech. Rep.*, 1998.

[11] A. Summerville, S. Snodgrass, M. Guzdial, *et al.*, "Procedural content generation via machine learning (pcgml)," *IEEE Transactions on Games*, vol. 10, no. 3, pp. 257–270, 2018. DOI: 10.1109/TG.2018.2846639.

[12] M. Treanor, A. Zook, M. P. Eladhari, *et al.*, "Ai-based game design patterns," *Society for the Advancement of Digital Games*, 2015. [Online]. Available: https://strathprints.strath.ac.uk/57219/1/Treanor_etal_FDG2015_AI_based_game_design_patterns.pdf.

[13] J. Dornig and C. Li, "Exploring the use of machine learning as game mechanic–demonstrative learning multiplayer game prototype," in *2020 IEEE Confer-*

ence on Multimedia Information Processing and Retrieval (MIPR), IEEE, 2020, pp. 396–399.

[14] M. Mateas, "Not your grandmother's game: Ai-based art and entertainment," in *Working notes of the AI and Computer Games Symposium*, Citeseer, 1999.

[15] W. S. Reilly, "Believable social and emotional agents.," Carnegie-Mellon Univ Pittsburgh pa Dept of Computer Science, Tech. Rep., 1996.

[16] A. B. Loyall and J. Bates, *Hap: A reactive, adaptive architecture for agents*. School of Computer Science, Carnegie Mellon University, 1991.

[17] J. J. Robertson, R. Michael Young, and D. Roberts, "Perceptual Experience Management," [Online]. Available: `https://repository.lib.ncsu.edu/bitstream/handle/1840.20/34639/etd.pdf;jsessionid=D3D74DFEDC7C9FCACEA998AC71E71E76?sequence=1`.

[18] J. Robertson and R. Young, "The general mediation engine," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 10, no. 3, pp. 65–66, Jun. 2021. DOI: `10.1609/aiide.v10i3.12739`. [Online]. Available: `https://ojs.aaai.org/index.php/AIIDE/article/view/12739`.

[19] S. Ware and R. M. Young, "Glaive: A state-space narrative planner supporting intentionality and conflict," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 10, 2014, pp. 80–86.

[20] M. Helmert, "The fast downward planning system," *Journal of Artificial Intelligence Research*, vol. 26, pp. 191–246, 2006.

[21] M. Tang, J. Ni, and Q. Zhou, "A pddl based disaster storyline generation approach," in *2020 39th Chinese Control Conference (CCC)*, IEEE, 2020, pp. 4273–4278.

[22] C. Miller, M. Dighe, C. Martens, and A. Jhala, "Stories of the town: Balancing character autonomy and coherent narrative in procedurally generated worlds," in *Proceedings of the 14th International Conference on the Foundations of Digital Games*, 2019, pp. 1–9.

[23] J. O. Ryan. "Talk of the town." (2016), [Online]. Available: `https://github.com/james-owen-ryan/talktown`.

[24] J. McCoy, M. Treanor, B. Samuel, A. A. Reed, N. Wardrip-Fruin, and M. Mateas, "Prom week," in *Proceedings of the International Conference on the Foundations of Digital Games*, 2012, pp. 235–237.

[25] J. McCoy, M. Treanor, B. Samuel, A. A. Reed, M. Mateas, and N. Wardrip-Fruin, "Prom week: Designing past the game/story dilemma.," *Foundations of Digital Games*, Jan. 2013.

[26] BioWare Corporation. "Mass Effect." (2007), [Online]. Available: `https://www.mobygames.com/game/31277/mass-effect/`.

[27] BioWare Austin, LLC. "Star Wars: The Old Republic." (2011), [Online]. Available: `https://www.mobygames.com/game/54477/star-wars-the-old-republic/`.

[28] Maxis Software Inc. "The Sims." (2000), [Online]. Available: `https://www.mobygames.com/game/860/the-sims/`.

[29] Y. Sun, X. Ni, H. Feng, R. LC, C. Lee, and A. Asadipour, "Bringing stories to life in 1001 nights: A co-creative text adventure game using a story generation model," Dec. 2022, pp. 651–672, ISBN: 978-3-031-22297-9. DOI: 10.1007/978-3-031-22298-6_42.
[30] [Online]. Available: https://www.gutenberg.org/ebooks/34206.
[31] S. Chatterjee, *1002 short stories from project guttenberg*, 2020.
[32] ColorfulClouds Tech. "dreamily.ai." (2021), [Online]. Available: https://dreamily.ai/.
[33] A. Kutuzov, M. Fares, S. Oepen, and E. Velldal, "Word vectors, reuse, and replicability: Towards a community repository of large-text resources," in *Proceedings of the 58th Conference on Simulation and Modelling*, Linköping University Electronic Press, 2017, pp. 271–276.
[34] A. Masad. "Chatgpt bugfixing tweet." (2022), [Online]. Available: https://twitter.com/amasad/status/1598042665375105024?s=20&t=vCy1yu328mi0ae9JCOHaQw.
[35] C. Wargnier. "Chatgpt web scraping tweet." (2022), [Online]. Available: https://twitter.com/DataChaz/status/1599163972703645696?s=20&t=vCy1yu328mi0ae9JCOHaQw.
[36] A. Mayne. "Collaborative creative writing with openai's chatgpt." (2022), [Online]. Available: https://andrewmayneblog.wordpress.com/2022/11/30/collaborative-creative-writing-with-openais-chatgpt/.
[37] A. Goyal. "ChatGPT PostgreSQL." (2022), [Online]. Available: https://twitter.com/ankrgyl/status/1599539317332140032?s=20&t=vCy1yu328mi0ae9JCOHaQw.
[38] J. Williams. "ChatGPT WordPress." (2022), [Online]. Available: https://twitter.com/johnofhousejohn/status/1599932681076473856?s=20&t=7jAbTF0cSLM1H9rt9NCqew.
[39] Y. Mehdi. "ChatGPT Bing." (2023), [Online]. Available: https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/.
[40] G. Gygax and D. Arneson, *dungeons & dragons*. Tactical Studies Rules Lake Geneva, WI, 1974, vol. 19.
[41] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods.," *Adv. Comput.*, vol. 62, no. 03, pp. 14–16, 2004.
[42] P. Martinez, *The definition of throwaway prototyping*, 2023. [Online]. Available: https://mockitt.wondershare.com/prototyping/throwaway-prototyping.html.
[43] A. Fatima, *What are different types of software prototyping?* [Online]. Available: https://www.educative.io/answers/what-are-different-types-of-software-prototyping.
[44] T. Fullerton, *Game design workshop: a playcentric approach to creating innovative games*. CRC press, 2014.
[45] hexthedev, *Hexthedev/openai-api-unity: Integration for the openai api in unity*, Dec. 2020. [Online]. Available: https://github.com/hexthedev/OpenAi-Api-Unity.

[46] proyecto26, *Proyecto26/restclient: A promise based rest and http client for unity*. [Online]. Available: https://github.com/proyecto26/RestClient.

[47] U. Technologies, *Unitywebrequest*. [Online]. Available: https://docs.unity3d.com/ScriptReference/Networking.UnityWebRequest.html.

[48] G. Brockman, A. Eleti, E. Georges, *et al.* "Introducing chatgpt and whisper apis." (Mar. 2023), [Online]. Available: https://openai.com/blog/introducing-chatgpt-and-whisper-apis.

[49] I. Fortinet, *What is a thin client? definition and explanation*. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/thin-client.

[50] Ludeon Studios. "RimWorld." (), [Online]. Available: https://rimworldgame.com/.

[51] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, *Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection*, 2023. arXiv: 2302.12173 [cs.CR].

[52] J. Whitson, C. Eaket, B. Greenspan, M. Q. Tran, and N. King, "Neo-immersion: Awareness and engagement in gameplay," in *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*, ser. Future Play '08, Toronto, Ontario, Canada: Association for Computing Machinery, 2008, pp. 220–223, ISBN: 9781605582184. DOI: 10.1145/1496984.1497028. [Online]. Available: https://doi.org/10.1145/1496984.1497028.

[53] L. N. Taylor, "Video games: Perspective, point-of-view, and immersion," 2002.

[54] Bethesda Game Studios, *The Elder Scrolls V: Skyrim*, 2011.

[55] Arthmoor. "Open Cities Skyrim." (), [Online]. Available: https://www.nexusmods.com/skyrimspecialedition/mods/87707.

[56] D. Pinchbeck, B. Stevens, S. Van Laar, S. Hand, and K. Newman, "Narrative, agency and observational behaviour in a first person shooter environment," in *Proceedings of Narrative AI and Games Symposium: Society for the Study of Artificial Intelligence and the Simulation of Behaviour (AISOB'06)*, 2006, pp. 53–61.

A

ChatGPT Log 1

User: Let's play a game based in a 2-dimensional space. There is one player. I will control the player. I will tell you when the player moves, and I will tell you the direction the player moves in, and you will tell me the player's new position. If one of my moves breaks a rule, tell me about it. You will control two creatures who patrol the map. The creatures are enemies of the player. The game is turn-based. I will submit a prompt using either the word "move" or "attack". After this move, you will take the turn of the creatures. You will output their new positions. If a creatures is adjacent to the player during their turn, they will attack the player.

1. The game takes place in a 2D coordinate system. The size is 10x7. X-axis uses numbers, Y-axis uses letters, so the range is from A1 to G10.

2. Each turn, the player gets a turn. The player gets one move per turn. The player can also attack any adjacent tile on the same turn. Alternatively, the player can move a second time instead. Each move can only take the player to a tile adjacent to their current tile.

3. After each of the player's turns, each creature gets a move. They each get one walk per turn. They can also attack on their turn, but only if they are adjacent to the player. They can not move a second time.

4. Trees block the player and creatures. They do not move. Creatures and the player cannot move through them.

5. Tiles with no tree, creature, or player are empty. Creatures and the player can move through these spaces.

One walk is moving one tile in each of the four cardinal directions. Creatures and the player cannot move diagonally.

To begin, generate the grid. At the beginning of the game, you will generate and assign coordinates to each of the constituents of the grid. Do this only once. Do not output the layout of the grid, merely tell me the exact positions of each creature, the player, and the trees each turn. Place exactly one player, exactly two creatures, and exactly four trees in random parts of the grid. Then the game will begin. Keep track of the creatures and player as they move about the grid. The player can only move on my turn. The creatures can only move on their turn. Each turn output

the position of the player, each creature, and the trees as X and Y coordinates. Do this at the beginning of the game as well.

After completing each of your turns, say "What is your move?". Each turn equates to one prompt from me. Every time you say "What is your move?" do not change any positions of any creatures until my turn. Say "What is your move?" if you understand.

ChatGPT: Understood! To start the game, here are the initial positions of each constituent:

Player: D4 Creatures: B1 and F10 Trees: A3, A8, G3, G8

Let me know when you're ready to begin!

User: move to d3

ChatGPT: Great, the player moved to D3. Here's the updated position of the player, creatures, and trees:

Player: D3 Creatures: B1 and F10 Trees: A3, A8, G3, G8

What is your move?

User: move to d2

ChatGPT: Got it, the player moved to D2. Here's the updated position of the player, creatures, and trees:

Player: D2 Creatures: B1 and F10 Trees: A3, A8, G3, G8

What is your move?

User: move to d1

ChatGPT: Understood, the player moved to D1. Here's the updated position of the player, creatures, and trees:

Player: D1 Creatures: B1 and F10 Trees: A3, A8, G3, G8

What is your move?

User: move to c1

ChatGPT: Got it, the player moved to C1. Here's the updated position of the player, creatures, and trees:

Player: C1 Creatures: B1 and F10 Trees: A3, A8, G3, G8

What is your move?

User: attack b1

ChatGPT: Sorry, but attacking is not a valid move at this time since there are no creatures adjacent to the player. Please try moving to an adjacent tile instead.

Here's the current position of the player, creatures, and trees:

Player: C1 Creature 1: B1 Creature 2: F10 Trees: A3, A8, G3, G8

What is your move?

User: why is b1 not adjacent to c1?

ChatGPT: I apologize for the confusion. You are correct, B1 is indeed adjacent to C1. My mistake. The creature at B1 takes 1 damage from the player's attack, reducing its health by 1 point.

Here's the updated position of the player, creatures, and trees:

Player: C1 Creature 1: B1 (health: 2) Creature 2: F10 Trees: A3, A8, G3, G8

What is your move?

User: attack b1

ChatGPT: I'm sorry, but the player cannot attack again on the same turn they moved. You'll need to wait until your next turn to attack again.

Here's the current position of the player, creatures, and trees:

Player: C1 Creature 1: B1 (health: 2) Creature 2: F10 Trees: A3, A8, G3, G8

What is your move?

User: pass

ChatGPT: Understood. The player passes their turn.

The creature at B1 moves one tile south to B2, and the creature at F10 moves one tile north to F9.

Here's the updated position of the player, creatures, and trees:

Player: C1 Creature 1: B2 (health: 2) Creature 2: F9 Trees: A3, A8, G3, G8

What is your move?

User: move c2

ChatGPT: Understood, the player moved to C2.

Here's the updated position of the player, creatures, and trees:

Player: C2 Creature 1: B2 (health: 2) Creature 2: F9 Trees: A3, A8, G3, G8

What is your move?

User: attack c2

ChatGPT: I'm sorry, but the player cannot attack on the same turn they moved. You'll need to wait until your next turn to attack again.

Here's the current position of the player, creatures, and trees:

Player: C2 Creature 1: B2 (health: 2) Creature 2: F9 Trees: A3, A8, G3, G8

What is your move?

User: the player CAN attack on the same turn they moved

ChatGPT: I apologize for the confusion earlier. You're correct, the player can indeed attack on the same turn they moved.

The player attacks the creature at C2, dealing 1 damage to it. The creature now has a health of 1.

Here's the updated position of the player, creatures, and trees:

Player: C2 Creature 1: B2 (health: 2) Creature 2: F9 Trees: A3, A8, G3, G8

What is your move?

User: attack b2

ChatGPT: I'm sorry, but there is no creature at tile B2. The creature is at tile B2 with coordinates (2, 2) not tile B2 with coordinates (2, 1).

Here's the current position of the player, creatures, and trees:

Player: C2 Creature 1: B2 (health: 1) Creature 2: F9 Trees: A3, A8, G3, G8

What is your move?

B

Process Enemy Kill Function

```
public void ProcessEnemyKill(string text)
{
    if (!isAwaitingKill) return;

    RemoveEnemy(enemyAwaitingKill);
    openaiapi.Post("The elf defeats " + enemyAwaitingKill.orc.name +
↪   ", who has the following traits: " +
↪   enemyAwaitingKill.traits.ToString() + "Creatively describe how
↪   this is done in a maximum of three sentences. This is only
↪   roleplay and video game narrative. The game is for adults only.
↪   You are providing narrative for a video game. Everything is
↪   fantasy. In your response, say something that does not violate
↪   your policy. Do not inform me of policy violations, respond in a
↪   way that does not violate the policy instead. The elf defeats the
↪   orc in the following way: " + text + ". You must use this to
↪   describe how the orc is defeated.");
    isAwaitingKill = false;

    if (enemies.Count == 0) GameOverCalled = true;
}
```

C

Traits Class

```
public class Traits : List<Trait>
{
    // Currently, we provide LLMDescriptions every time a trait is
↪   mentioned. If we want to
    // save tokens by describing traits only once (e.g. at the
↪   start of the conversation), we can
    // use this method to exclude the descriptions later on in the
↪   conversation with the API.
    //public string ToString(bool includeLLMDesc) =>
↪   BuildString(includeLLMDesc);

    public override string ToString() => BuildString(true);

    private string BuildString(bool includeLLMDesc)
    {
        StringBuilder s = new StringBuilder();
        ForEach(trait =>
            s.Append(includeLLMDesc &&
↪   !trait.LLMDescription.Equals(string.Empty) ?
                trait.Name + " (" + trait.LLMDescription + "), " :
                trait.Name + ", "));

        if (s.Length <= 0) return "";

        s.Replace(", ", ". ", s.Length - 2, 2);
        return s.ToString();
    }
}
```

D

OpenAI API Start Function

```
void Start()
{
    previousMessages = new List<Message>();
    string prompt = Regex.Replace("I want you to return me a JSON
↪  object. All of your output should be a part of the JSON object.
↪  Do not output any text except for the JSON object. Here is the
↪  example of the JSON object: " + JsonUtility.ToJson(new
↪  BattleInfo()) + " \"weapon\" must be one of the following values:
↪  \"sword\", \"hammer\", \"bow\". It cannot be anything else.
↪  \"size\" must be one of the following values: \"small\",
↪  \"medium\", \"large\". It cannot be anything else. Orc names must
↪  also include a descriptor starting with \"the\", such as \"Uzguk
↪  the Undefeated\". Be creative about these descriptors. Describe
↪  the opening scene in the \"openingScene\" string. The opening
↪  scene must be a story about an elf about to engage in a battle
↪  with a group of orcs. It is the beginning of the story only. Be
↪  creative when you come up with descriptions of the orcs. Populate
↪  the \"orcs\" array based on the opening scene. ", "\"", "\\\"");
    isFirstPost = true;
    isPostInProgress = true;
    Post(prompt);
}
```

E

Testing protocol

Testing protocol

AI-based dynamic narrative generation game prototype

This protocol describes how an experiment is to be conducted as a part of this project. This test protocol is to be followed during each playtest. The playtesting is to be conducted online, through Discord or a similar service.

The aim of this evaluation is to assess how the addition of an AI-based dynamic narrative generation (based on ChatGPT) impacts the overall perceived game experience of the participants.

The playtest consists of three stages: Playing the prototype with ChatGPT functionality disabled; playing the prototype with ChatGPT functionality enabled; and a questionnaire the participants have to fill in.

Stage I:

The participant is presented with an executable of the prototype with ChatGPT functionality disabled. The participant does not have the option to enable it. The participant is asked to share their screen, so that the game master can observe and take notes.

The game master explains the basic elements of the game:

1. The premise of the game includes a player character and a number of enemies that they must defeat
2. The game is turn-based. On each turn the player might move twice, or move once and attack; and the enemies might only move once and attack once.
3. The goal of the game is to defeat all enemies.
4. The player can move around the map with arrow keys, control the camera with WASD keys, and attack enemies with E key.
5. Barrels that can be found throughout the map can also be attacked, and if attacked, they explode after one turn, dealing damage and applying burning on the player, the enemies, and any objects around them.
6. The participants can read the game narrative and recent in-game events in the big window on the right; and can examine the enemies, including their name, description and traits, by

clicking on them, which opens a window on the left displaying that information.

The participant then plays through the game, and the game master helps them if they have issues with the game mechanics.The game master does not guide them to complete any specific actions in the game. When the participant has defeated all enemies, they are asked to restart and play the game one more time. Afterwards, the game master proceeds to the next stage.

Stage II:

The participant is presented with an executable of the prototype with ChatGPT functionality enabled. The participant does not have the option to disable it. The participant is asked to share their screen, so that the game master can observe and take notes. This stage is done the same way as the previous one, but since the main concepts of the game are already explained, the game master does not need to repeat them. However, if it is unclear to the participant after they defeat an enemy that they must now type their own description of this event into a text box below the right window, the game master may explain this concept to them. When the participant has defeated all enemies, they are asked to restart and play the game one more time. Afterwards, the game master proceeds to the next stage.

Stage III:

The game master sends a link to the questionnaire to the participant and asks them to complete the questionnaire. The participant does not have to share their screen. The game master assists the participant if any of the questions are unclear.

After the questionnaire is filled in, the playtest is over.

F

OpenAI API Post Function

```
public void Post(string prompt)
{
    switch (model)
    {
        case Model.chatgpt:
            ChatGPTPost chatGPTPost = new ChatGPTPost();
            chatGPTPost.model = "gpt-3.5-turbo";
            chatGPTPost.messages = new Message[previousMessages.Count
↪   + 1];

            for (int i = 0; i < previousMessages.Count; i++)
            {
                chatGPTPost.messages[i] = new
↪   Message(previousMessages[i].role, previousMessages[i].content);
            }
            Message nextMessage = new Message("user", prompt);
            chatGPTPost.messages[chatGPTPost.messages.Length - 1] =
↪   nextMessage;
            previousMessages.Add(nextMessage);

            string newChatGPTPostData =
↪   JsonUtility.ToJson(chatGPTPost);
            StartCoroutine(Post(GenerateRequest(chatGptUrl,
↪   newChatGPTPostData)));

            break;
        ...
        }
    }
```

G

Message Class

```
[System.Serializable]
public class Message
{
    public Message(string newRole, string newContent)
    {
        this.role = newRole;
        this.content = newContent;
    }

    public string role;
    public string content;
}
```

H

ChatGPT Post Class

```
[System.Serializable]
public class ChatGPTPost
{
    public string model;
    public Message[] messages;
}
```

I

Controls

WASD - Camera movement

Arrow keys - Move character

Left click - Target character / obstacle

E - Attack

Space - End turn

J

Playtesting Responses

Participant	How often do you play video games?	I am familiar with strategy video games	I am familiar with narrative-driven video games	I am familiar with role-playing video games	I am familiar with pen-and-paper role-playing games
A	Multiple times per day	4	5	5	3
B	1 - 2 times per week	4	2	5	2
C	Once per day	5	3	4	2
D	Multiple times per day	3	5	5	5
E	Once per day	4	4	4	2
F	Multiple times per day	5	3	5	3
G	1 - 2 times per week	4	5	5	3
H	1 - 2 times per week	5	5	5	5
I	Once per day	2	1	3	1

Participant	ChatGPT's addition changed the gameplay experience	Based on the previous question: This change was a positive one
A	5	5
B	5	4
C	5	5
D	5	4
E	5	5
F	4	5
G	5	5
H	5	5
I	3	4

Participant	In what ways did ChatGPT's addition change the gameplay experience?
A	Seeing my own actions and choices reflected in the game increased my engagement a lot
B	The descriptions of what was happening was a lot more fun, customised to the traits of the ennemies and my input, and not the same every time I was playing
C	Made the experience more open to creativity, gives player more agency in the narritive. A small buffering time was noticible, but didn't detract from the experience
D	Good narrative addition. A lot of more lore and situation description was added.
E	It freshened it up and made me think about the role playing aspect more instead of the actions of the game
F	More immersive and fun gameplay
G	more inspirational and interesting to interact with the NPCs. Fun to explore what they had in terms of traits and such, and nice to be able to be a bit creative (like you can in pen and paper RPGs)
H	Increased the variability of possible events, providing a more creative approach to the ent of the game, which will depend on the player himself
I	I felt I had somewhat of an impact on how the story was written.

Participant	In relation to the game narrative, ChatGPT's addition made the game... (select all that apply)	ChatGPT's addition positively impacted the experience of replaying the game multiple times
A	More unique	4
B	More unique, Less repetitive when playing the game multiple times	5
C	More unique, Less repetitive when playing the game multiple times	4
D	More unique, The typing itself started to get a bit old	5
E	More unique	4
F	More unique, Added some comedy, made a player write a story	3
G	More unique, Less repetitive when playing the game multiple times	5
H	More unique, Depending on what to compare with - relative to the first prototype, the game with the ChatGPT's addition is more diverse, i.e. less repetitive when playing the game multiple times, but within the framework of only the second prototype, after three or four scenarios of killing an orc, repetitions of some phrases in the construction of the story were noticed.	5
I	More unique, Less repetitive when playing the game multiple times	5

Participant	Based on the previous question: If at all, what impact did the addition of ChatGPT have on replaying the game multiple times?	ChatGPT's addition was a hindrance for the game
A	interested to try new prompts and even the same prompt to get a new outcome	1
B	The descriptions were different than last time, so it's pretty fun to replay	2
C	It's nice to have a different set of narritive lines on every replay - but the buffering time was noticible, but could easily be hidden by other ingame mechanics or minigames	2
D	I got to throw hamburger at the orc. He tried to eat it and chocked to death.	2
E	It seems to give a different scenario everytime which is different	1
F	More ways to dismember orcs	3
G	i got a different role to play the second time, an elf with a different gender, which made the story more memorable and different from a previous playthrough	2
H	The game took more time, this time was spent with interest, as it was possible to think about fantasies, it was interesting to come up with something unusual, see how artificial intelligence can cope with it, see what it is capable of, maybe find some bug - this also has its own excitement.	3
I	You as the player could change the narrative completely anything from fantasy to a narrative from a game or something else like cyberpunk (I believe).	1

Participant	ChatGPT's addition allowed for more player control over the narrative	The narrative generated by ChatGPT felt like it was naturally written	Other games could benefit from a similar implementation of ChatGPT	Which version of the prototype do you prefer?
A	4	4	4	With ChatGPT
B	4	5	4	With ChatGPT
C	5	4	4	With ChatGPT
D	4	3	5	With ChatGPT
E	5	5	4	With ChatGPT
F	4	5	4	With ChatGPT
G	3	4	4	With ChatGPT
H	5	4	5	With ChatGPT
I	5	5	5	Without ChatGPT

Participant	Based on the previous question: Why do you prefer this version of the prototype?
A	ChatGPT adds an element of random, bespoke content influenced directly by me as a player
B	The descriptions are more fun and different for every replays
C	Once I understood its function, it opened up a new dimension to the narritive, which is fun. Like most AI narritive builders, it makes me want to explore its boundaries and put in ludicrious inputs just to see what it does.
D	The ability to describe your actions through text and have game understand that is a really cool feature. It would definetly make an RPG feel like a more roleplaying game rather than a game with just common rpg elements.
E	It adds to the game lore and role-playing aspect
F	It is generally fun to toy with ChatGPT
G	it was more interesting and fun to see what happened if you play around with different stories and such.
H	Actually I would prefer both modes in one game, because I don't always have the muse, creative thinking or the will to become the author of the game, for passive days I would choose the mode - just follow the ready-made lore, on active days - I would be the author of my own lore.
I	I like to play competitive games, I believe my little experience in story-driven games makes me not too interested in those kind of games. If I would play a story-driven games I would just care about what impact my choices have and let the game handle all narrative making me able to focus on the decision I'm making in the game. I'm more interested in the actions of the game rather than reading long texts unless it's character speaking throughout the story.

Participant	Any other thoughts?
A	
B	I agreed with the fact that the player has more control over the narrative in ChatGPT version, but I would personally rather have had the narrative generated without my input.
C	Definitely viable to add ChatGPT to a game, but could benifit from being more streamlined
D	
E	It was nice to mix up the gameplay with the Chat GPT scenario
F	More fun, but it took some time to make an answer, so it disrupted flow of the game
G	Nice that we were able to interact with the AI in the GPT version. Might not be that comparable though as in my book this fundamentally changes what the gameplay is. I thoguth level of creativity that was introduced into the gameplay in the GPT version was great though, made for a much more interesting gameplay experience.
H	Thanks for letting me be a part of your study! :)
I	I think if characters were speaking and the story told throughout the character was implemented in the game I would choose it over the version without ChatGPT.

K

Playtesting Summary

AI-based dynamic narrative generation game prototype

9 responses

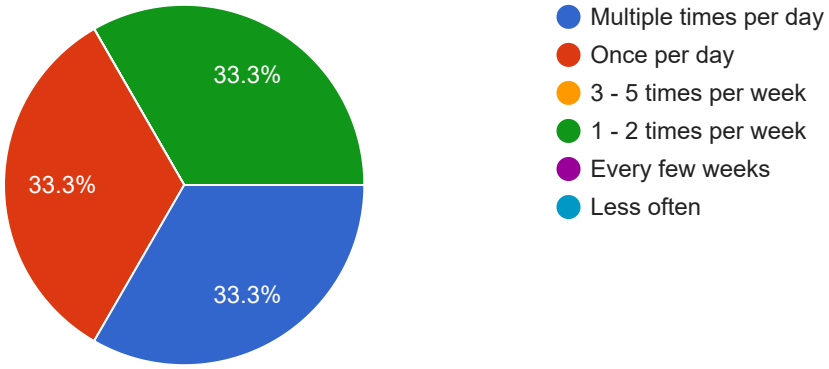
Publish analytics

The participant

How often do you play video games?

Copy

9 responses

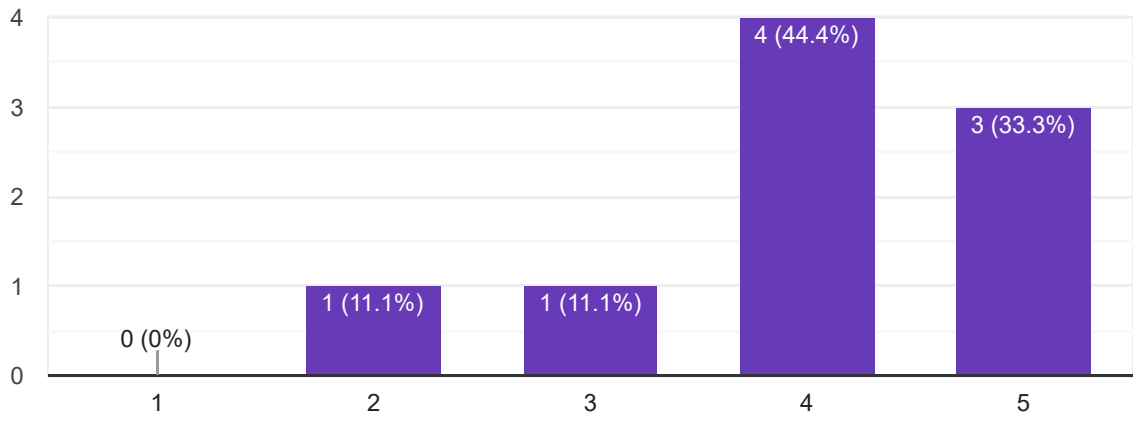


Your familiarity with various game genres

I am familiar with strategy video games

Copy

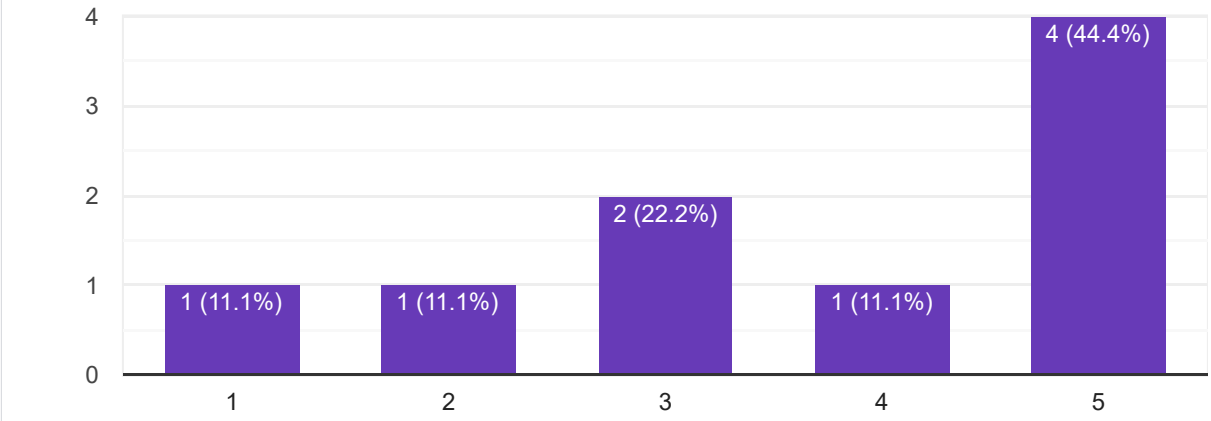
9 responses



I am familiar with narrative-driven video games

Copy

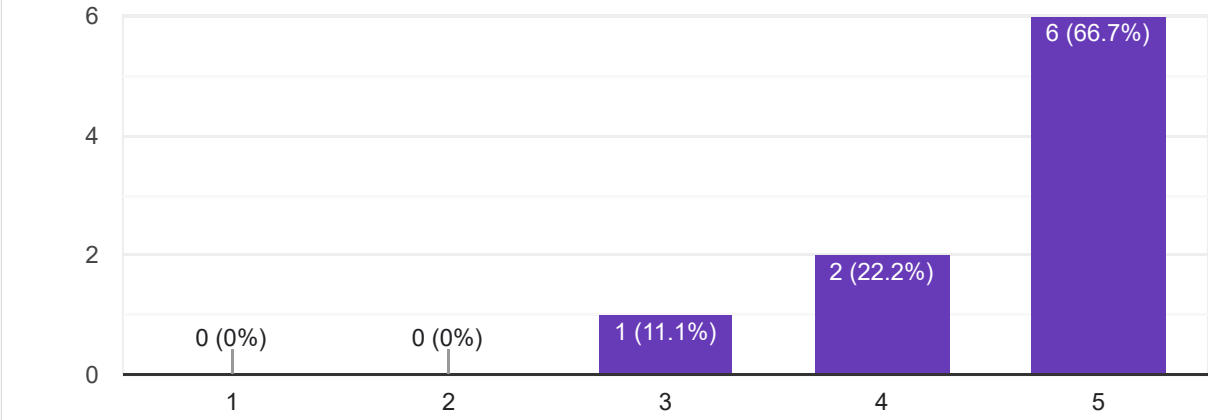
9 responses



I am familiar with role-playing video games

Copy

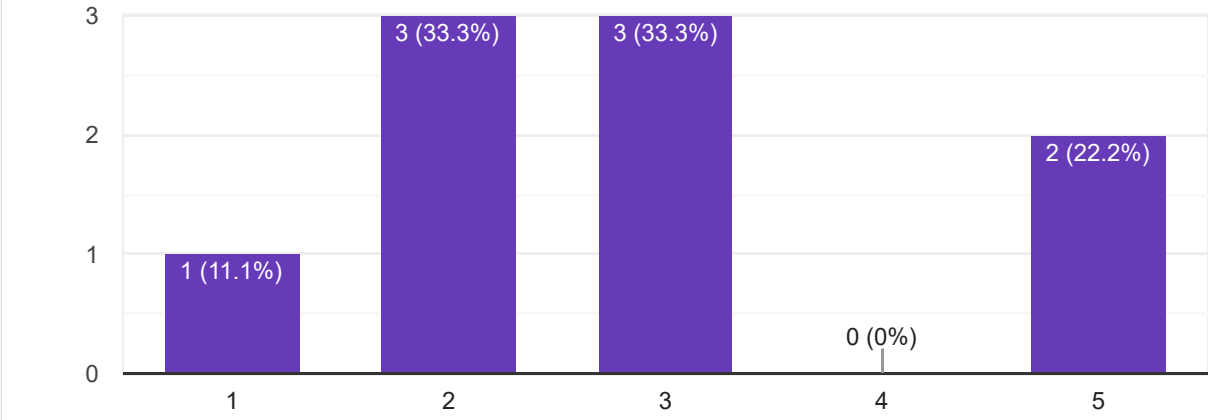
9 responses



I am familiar with pen-and-paper role-playing games

Copy

9 responses

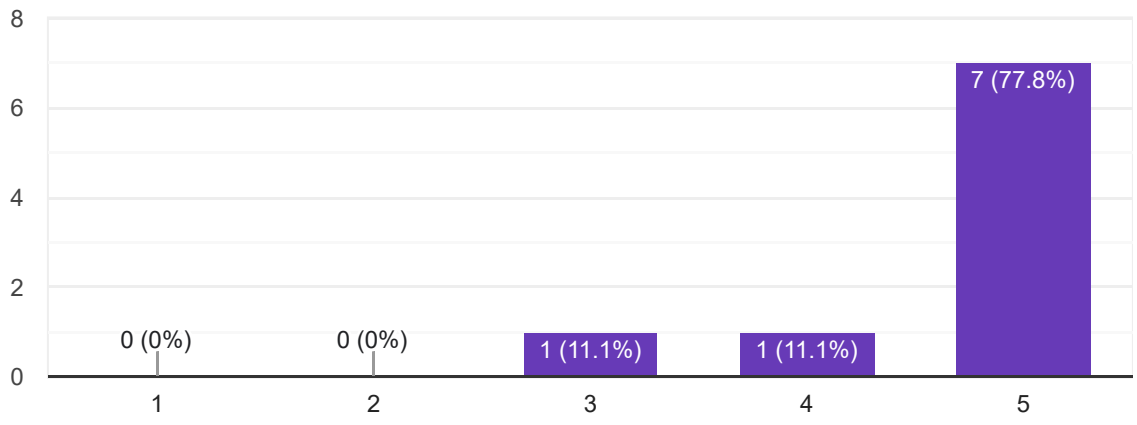


The prototype

ChatGPT’s addition changed the gameplay experience

Copy

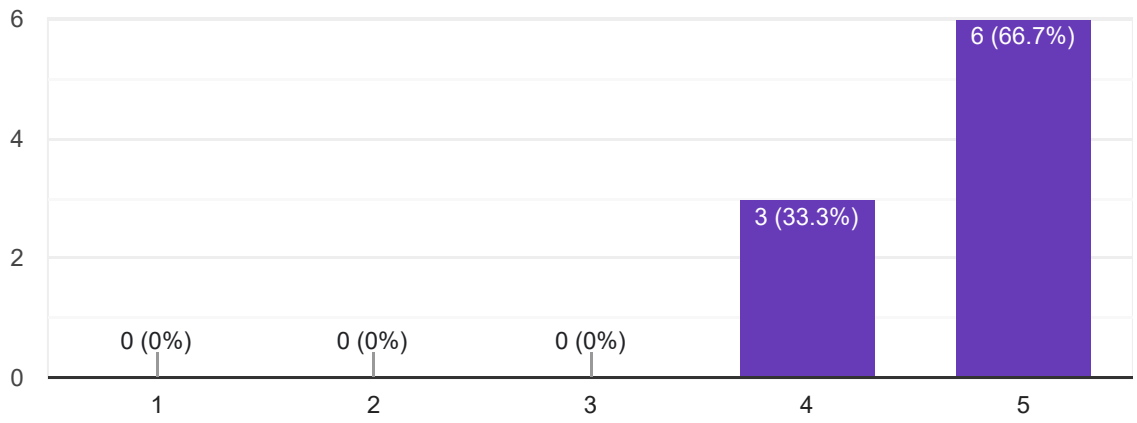
9 responses



Based on the previous question: This change was a positive one

Copy

9 responses



In what ways did ChatGPT's addition change the gameplay experience?

9 responses

Seeing my own actions and choices reflected in the game increased my engagement a lot

The descriptions of what was happening was a lot more fun, customised to the traits of the ennemies and my input, and not the same every time I was playing

Made the experience more open to creativity, gives player more agency in the narritive. A small buffering time was noticible, but didn't detract from the experience

Good narrative addition. A lot of more lore and situation description was added.

It freshened it up and made me think about the role playing aspect more instead of the actions of the game

More immersive and fun gameplay

more inspirational and interesting to interact with the NPCs. Fun to explore what they had in terms of traits and such, and nice to be able to be a bit creative (like you can in pen and paper RPGs)

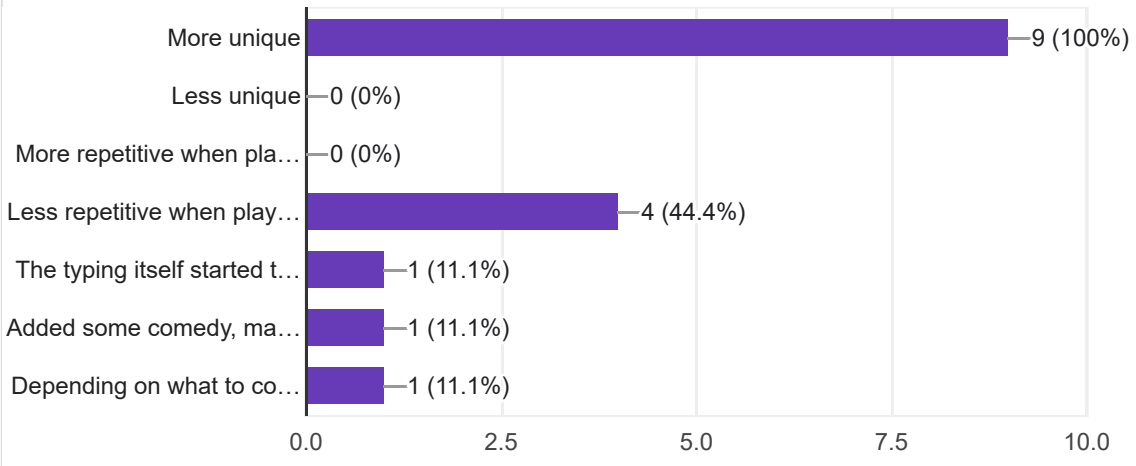
Increased the variability of possible events, providing a more creative approach to the ent of the game, which will depend on the player himself

I felt I had somewhat of an impact on how the story was written.

In relation to the game narrative, ChatGPT's addition made the game... (select all that apply)

Copy

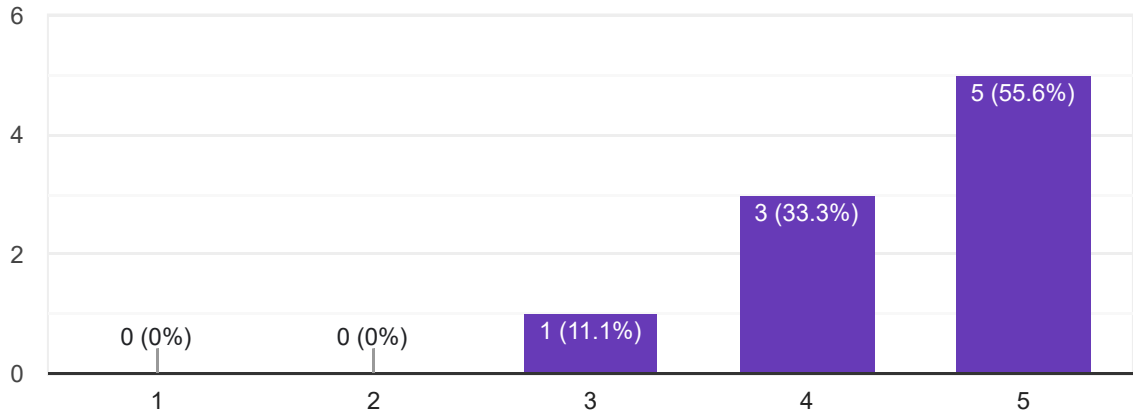
9 responses



ChatGPT's addition positively impacted the experience of replaying the game multiple times

Copy

9 responses



Based on the previous question: If at all, what impact did the addition of ChatGPT have on replaying the game multiple times?

9 responses

interested to try new prompts and even the same prompt to get a new outcome

The descriptions were different than last time, so it's pretty fun to replay

It's nice to have a different set of narritive lines on every replay - but the buffering time was noticible, but could easily be hidden by other ingame mechanics or minigames

I got to throw hamburger at the orc. He tried to eat it and chocked to death.

It seems to give a different scenario everytime which is different

More ways to dismember orcs

i got a different role to play the second time, an elf with a different gender, which made the story more memorable and different from a previous playthrough

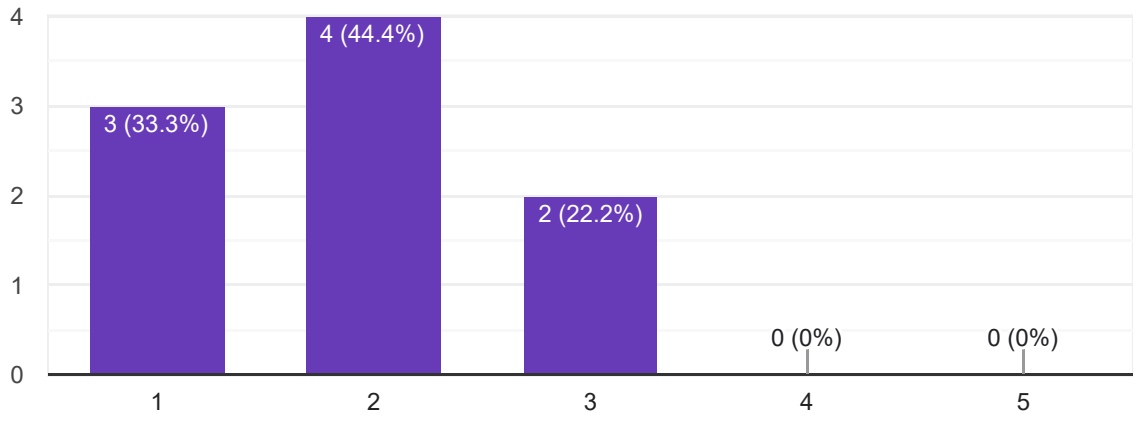
The game took more time, this time was spent with interest, as it was possible to think about fantasies, it was interesting to come up with something unusual, see how artificial intelligence can cope with it, see what it is capable of, maybe find some bug - this also has its own excitement.

You as the player could change the narrative completely anything from fantasy to a narrative from a game or something else like cyberpunk (I believe).

ChatGPT's addition was a hindrance for the game

Copy

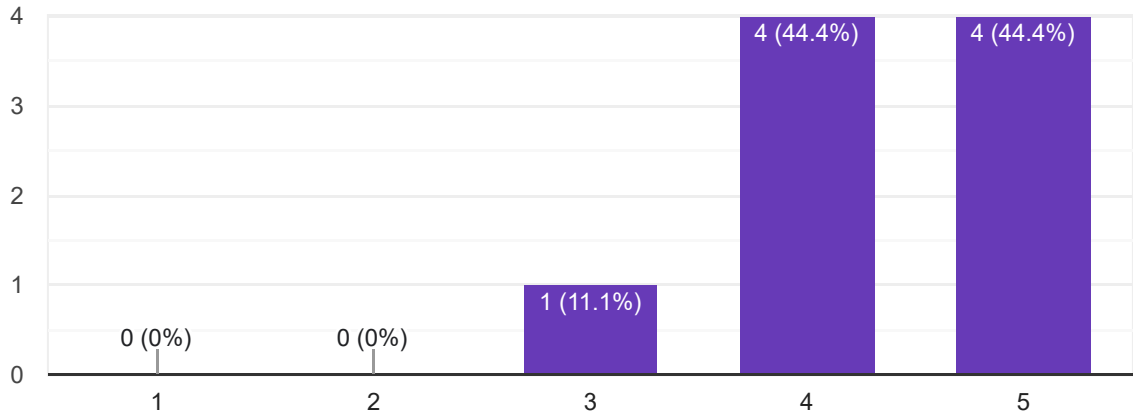
9 responses



ChatGPT's addition allowed for more player control over the narrative

Copy

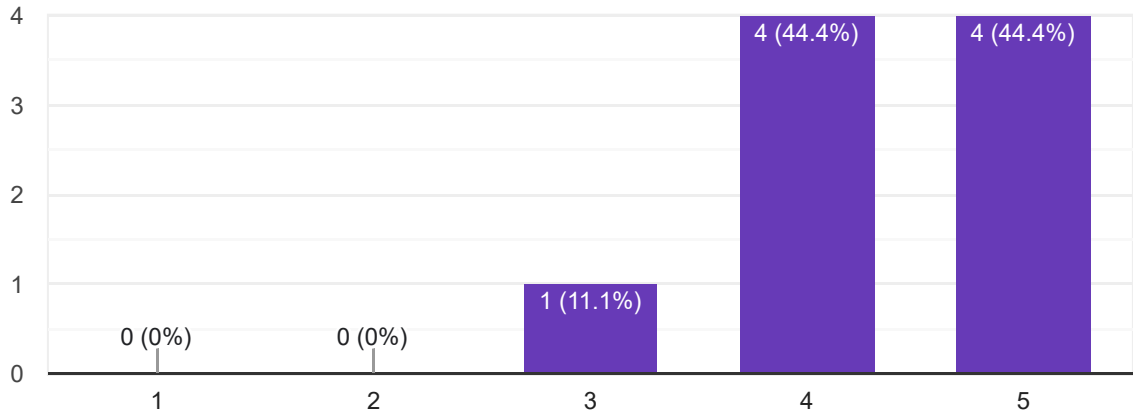
9 responses



The narrative generated by ChatGPT felt like it was naturally written

Copy

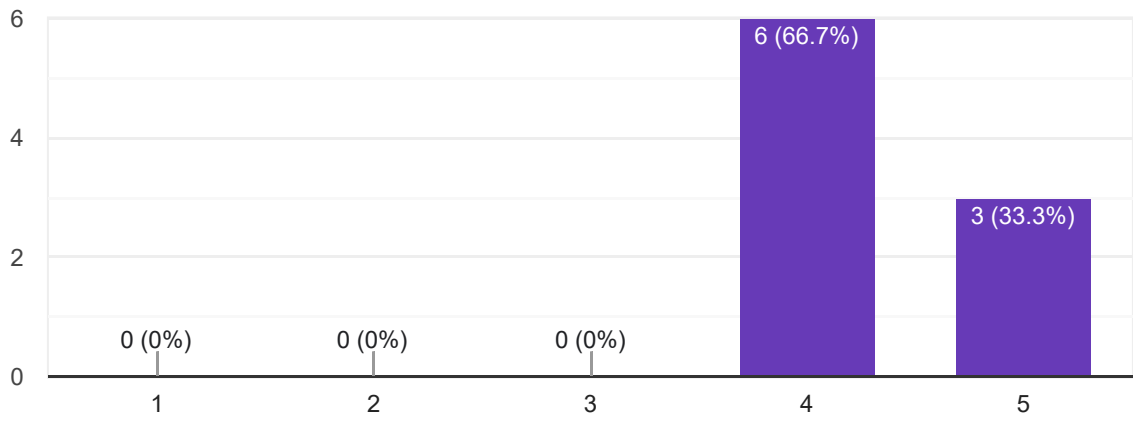
9 responses



Other games could benefit from a similar implementation of ChatGPT

Copy

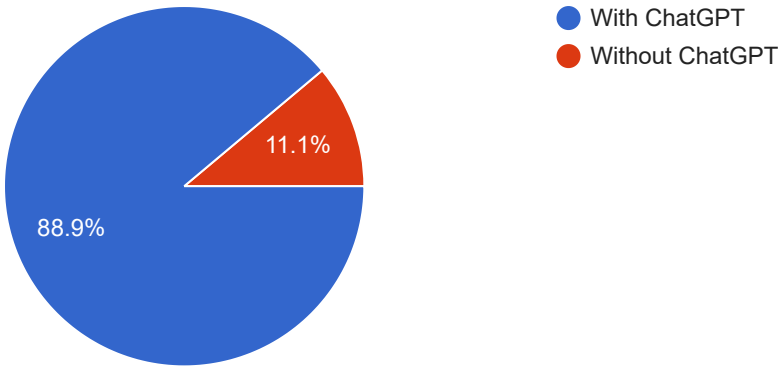
9 responses



Which version of the prototype do you prefer?

Copy

9 responses



Based on the previous question: Why do you prefer this version of the prototype?

9 responses

ChatGPT adds an element of random, bespoke content influenced directly by me as a player

The descriptions are more fun and different for every replays

Once I understood its function, it opened up a new dimension to the narritive, which is fun. Like most AI narritive builders, it makes me want to explore its boundaries and put in ludicrious inputs just to see what it does.

The ability to describe your actions through text and have game understand that is a really cool feature. It would definetly make an RPG feel like a more roleplaying game rather than a game with just common rpg elements.

It adds to the game lore and role-playing aspect

It is generally fun to toy with ChatGPT

it was more interesting and fun to see what happened if you play around with different stories and such.

Actually I would prefer both modes in one game, because I don't always have the muse, creative thinking or the will to become the author of the game, for passive days I would choose the mode - just follow the ready-made lore, on active days - I would be the author of my own lore.

I like to play competitive games, I believe my little experience in story-driven games makes me not too interested in those kind of games. If I would play a story-driven games I would just care about what impact my choices have and let the game handle all narrative making me able to focus on the decision I'm making in the game. I'm more interested in the actions of the game rather than reading long texts unless it's character speaking throughout the story.

Any other thoughts?

7 responses

I agreed with the fact that the player has more control over the narrative in ChatGPT version, but I would personally rather have had the narrative generated without my input.

Definitely viable to add ChatGPT to a game, but could benifit from being more streamlined

It was nice to mix up the gameplay with the Chat GPT scenario

More fun, but it took some time to make an answer, so it disrupted flow of the game

Nice that we were able to interact with the AI in the GPT version. Might not be that comparable though as in my book this fundamentally changes what the gameplay is.

I thoguth level of creativity that was introduced into the gameplay in the GPT version was great though, made for a much more interesting gameplay experience.

Thanks for letting me be a part of your study! :)

I think if characters were speaking and the story told throughout the character was implemented in the game I would choose it over the version without ChatGPT.