Enhancing Interactive Gaming with GPT-World

Advancing LLM-Powered Dynamic Content Creation and Logical Coherence

Maxwell Liu 23052907g@connect.polyu.hk

Abstract

This paper explores the application of Artificial Intelligence (AI) in dynamically generating and maintaining content in video games, and how these technologies can enhance the scalability of game design and player experience. I introduce a project named GPT-World, which utilizes the latest GPT-4 technology to create a rich, text-driven gaming environment where player decisions can significantly impact the development of the virtual world.

Research indicates that while games based on large language models (LLMs), such as AI Dungeon, have made progress in increasing interactivity, these games still suffer from issues of inconsistency and unclear logic. The innovation of the GPT-World project lies in its ability to generate both textual and visual content, and by quantifying game assets, it enhances the logical coherence and playability of game mechanics. Moreover, through a system based on JSON format, the LLM can access and integrate previous game content when necessary, ensuring story coherence and logical consistency.

The aim of this study is to bring innovation to the field of video games through AI technology, particularly by leveraging the generative capabilities of GPT-4, making game content not only responsive to player choices but also capable of adapting and expanding automatically, thereby providing a more immersive and personalized gaming experience. Looking forward, the application of AI in game design is expected to deepen, bringing more innovative possibilities.

Introduction

Games are defined as activities of problem-solving with an element of fun (Schell, 2020, p. 48). However, traditional fixed-content video games face the challenge of limited scalability, meaning the ability of game problems and plots to unfold in response to player decisions is constrained. To achieve dynamic matching and unlimited expansion of game content with player decisions, we need to leverage the creativity and understanding of Artificial Intelligence (AI) to generate and maintain new game assets and to provide feedback based on an understanding of player behavior.

Recently, GenAI technology (Muller et al., 2022) has made it possible to use AI to autonomously generate and maintain game content. Our project, GPT-World, employs GPT-4 technology (OPENAI, 2023) to create a rich, text-driven gaming environment, allowing players' actions to profoundly impact the virtual world. Generating such interactive narrative worlds requires meeting several criteria: everyday and thematic knowledge, semantic consistency, engaging content, overall coherence, and fluid, natural descriptions (Ammanabrolu et al., 2020).

Currently, interactive narrative games based on large language models (LLMs), such as AI Dungeon, face issues of inconsistency, unclear logic, and difficulty in control (Samuel et al., 2021; Gallotta et al., 2024). These issues highlight the existing methods' shortcomings in handling game narrative coherence and logical consistency, necessitating further research and improvement.

In this context, the innovation of the GPT-World project lies not only in its ability to generate textual and visual content but also in enhancing the logical coherence and playability of game mechanics through the quantification of game assets. Additionally, the LLM can access and integrate previous game content when necessary, ensuring story coherence and logical consistency. We propose a system that stores quantified information of entities generated by the LLM based on Class targets in JSON format, which can be automatically retrieved by the LLM when needed. This design supports credible logic and the expansion of gameplay depth based on quantified information in AI-driven games. Furthermore, the read-write system of asset information in the game environment helps maintain information linkage and consistency in the construction and description of the virtual world by the LLM.

Background and Motivation

Part 1: Evolution from AI-Base to AI-Native in the AI Game Field

Traditional "AI-Base" games have adopted methods such as evolutionary computation, reinforcement learning, and supervised learning (Yannakakis and Togelius 2014). For instance, Ammanabrolu et al. (2019) employed Markov chains and LSTM to generate quests in TextWorld; Van Stegeren and Myśliwiec (2021) trained GPT2 to generate NPC tasks for "World of Warcraft"; Sudhakaran et al. (2023) used PCG to generate levels for Super Mario Bros; Värtinen et al. (2024) explored using Quest-GPT-2 and untrained GPT-3 to generate game tasks.

Since 2023, the advancements in generative AI (GenAI) such as GPT4 (OPENAI, 2023) have facilitated the application and research of GenAI-based games (Gallotta et al., 2024). GenAI not only changes the way game assets are generated but also enhances players' freedom in actions and decisions through understanding and communicating content. As GenAI becomes more intelligent, AI is increasingly becoming an "AI-Native" core mechanism in games (Ceze 2022). For example, "1001 Nights" utilizes GPT4 and Stable Diffusion to drive the game, while in Yandere Simulator, players interact with LLM characters to complete tasks (Yan 2023). These innovations demonstrate the potential of AI in providing new gameplay experiences and highlight how developers ensure that AI-generated content is controllable while also enhancing the fun of games through AI's creativity.

Part 2: Potentials and Challenges of AI-Native Games, Using DnD as an Example

Among all forms of entertainment, games are the most affected by generative artificial intelligence (GenAI) because they are the most complex and interactive form of entertainment, emphasizing real-time experience (Gwertzman & Soslow, 2022). Particularly, Dungeons and Dragons (DnD) is considered an appropriate challenge for the next stage of artificial intelligence. DnD, which primarily relies on textual description, is well suited for current AI technologies (Ellis and Hendler, 2017; Louis and Sutton, 2018). Despite its long history, DnD remains popular and has seen a significant revival in both player communities and academic research in recent years (Scriven 2021).

Currently, various DnD tools aim to address difficulties in the game. For example, Avrae is a D&D Discord bot with over five million users that allows mathematical calculations using standard code formats (D&D Beyond 2022). D&D Beyond acts as a digital DnD tool that facilitates players' management and querying of characters, rules, etc. (D&D Beyond 2017). Kobold Fight Club helps players quickly create balanced encounters with monsters (Karl, 2010). These tools each focus on specialized functions such as quantitative calculations, game asset management, and game asset generation, but none yet combines all essential DnD functions.

For aspects requiring high logical and analytical capabilities, such as the narration and management by the Game Master (GM), the application of AI is considered a solution to problems that traditional digital tools cannot handle. Santiago (2023) explored various scenarios using large language models (LLMs) and other generative AI models to assist DMs, including alleviating cognitive load and providing inspiration. Additionally, by altering prompt design, researchers have demonstrated that GPT-3 can generate original and domain-specific content, showing that LLMs can creatively conceive D&D monsters using thematic knowledge from training (Zhu 2023). These studies validate the potential of LLMs in handling intelligent functions.

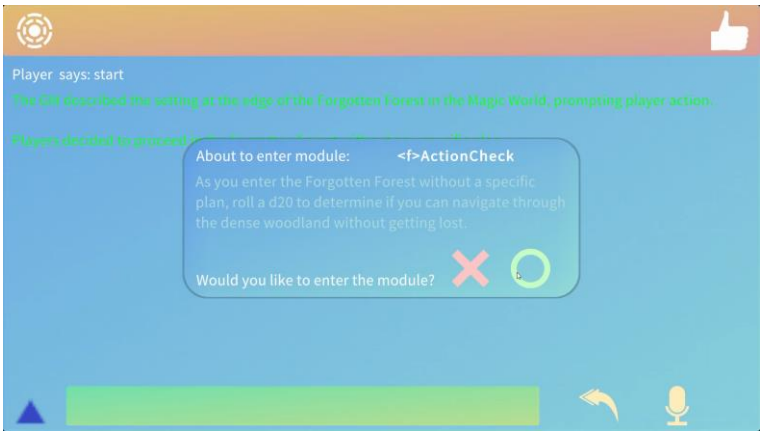
In terms of interactive narrative platforms aligned with the "AI-Native" concept, such as AI Dungeon (Latitude, 2020), Fables (Friends & Fables, n.d.), and HiddenDoor (Hidden Door, n.d.), these use LLMs to directly handle the game's narration, logic, and dialogue, allowing players' text inputs to directly intervene and alter the LLM's construction of the fictional world. However, due to AI's logical capabilities and token memory limits, these text-based games face issues with inconsistent generated content and unclear logic (Samuel et al. 2021; Gallotta et al. 2024). Referencing AI Dungeon's exploration at the forefront of "AI-Native," this game will focus on addressing these issues from both quantitative and automated read-write aspects, aiming to endow "AI-Native" games with the capability to handle complex game mechanics through mathematical calculations and the potential for long-term construction and operation.

Concept

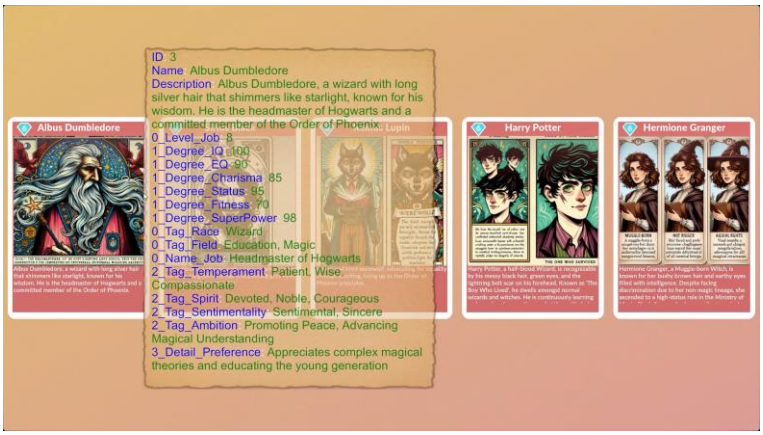
Story/narrative progression is the most significant aspect of gameplay in DnD, where constructing a grand, cohesive worldview, allowing the world to center around the player, and enabling the player's actions tohave long-term effects on the story are key factors that highlight the enjoyment and meaningfulness of the Story (Sidhu, 2024)(Salen 2003). The construction of a grand worldview and the long-term impact on the story require the narrative system to be consistently unified in space (content) and coherent over time. Quantifying and standardizing the world's assets are beneficial for scaling content and providing benchmarks for comparison. Some scholars believe it is crucial to find a balance between giving players the freedom to shape the story and adhering to the rules and settings of the game to ensure the experience remains immersive (Sun, 2023). Quantifying resources further facilitates the construction and use of game mechanics that process and utilize quantitative data, making the world's response to the player's actions more logical, conforming to digital scales and computational rules.

Game Play Process

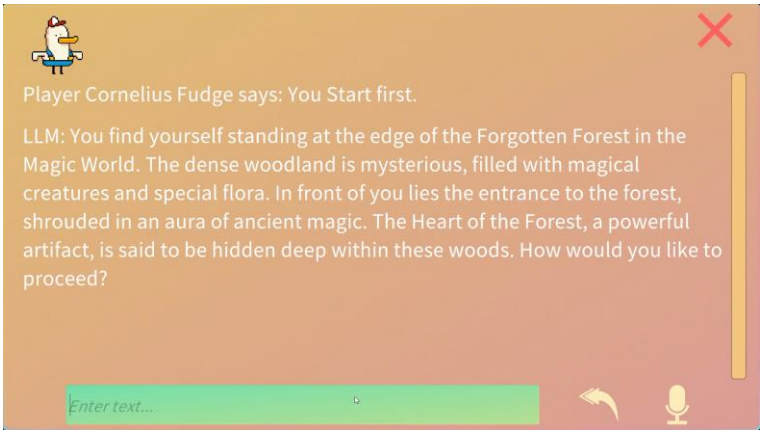
Module Selection: The player chooses whether to enter one of the modules: Playwright, Teller, Talker, or Checker.



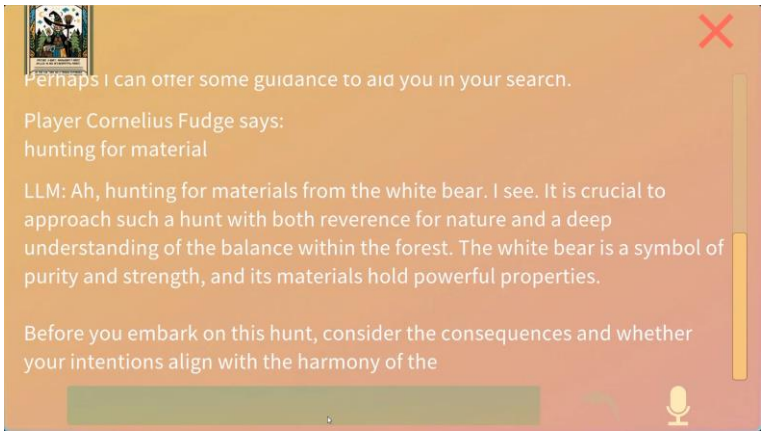
Playwright: Generates scripts, as well as the performers and scenes required to enact them.



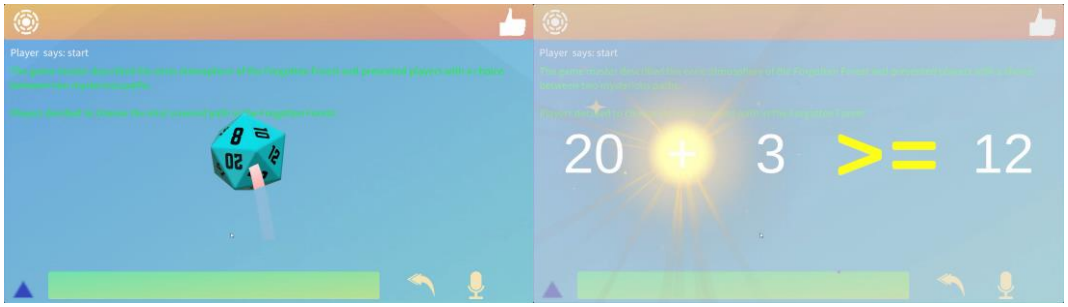
Teller: Describes scenes and encounters.



Talker: Engages in conversations. Players can gather information and influence relationships.



Checker: Checks whether the players' actions are successful.



After a successful action:

Update the information related to the actors.

Acquire skill cards that provide bonuses for similar actions in the future.

Continuing the adventure cycle: Generate new scripts, foresee new scenes, converse with new characters, or check new actions.

Technical Implementation

Part 1. JSON Asset Read/Write System and Module Division

1. Universal Template for Game Assets: Class_Base

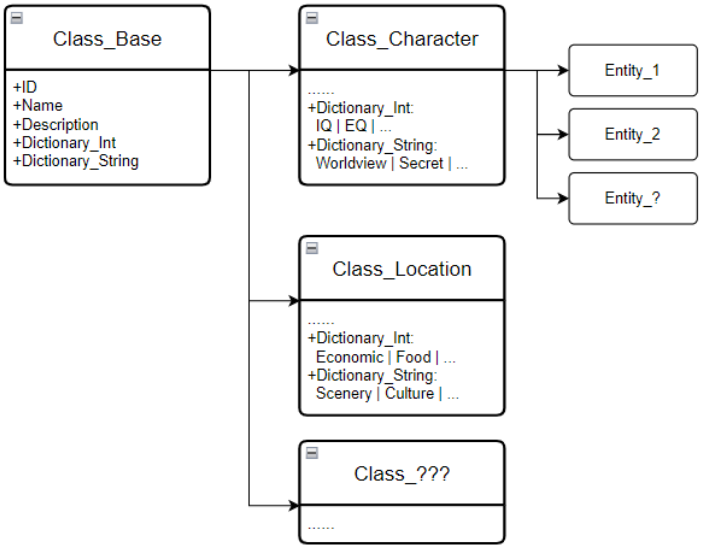
The diversity and complexity of games demand high flexibility and security in their functionality and logic. For this reason, Class_Base is introduced as a universal template for entities or quantified variable information to support game functions, such as the save system and row generation system. Serving as a superclass, Class_Base can support multiple subclasses (such as Class_Character, Class_Location, Class_Race, etc.) through a common function, thereby achieving code reuse and functional expansion.

Abstractly, each Entity (such as a specific character) is a carrier of multidimensional information. For example, Character A might possess attributes like high intelligence, low emotional intelligence, and a preference for art. Class_Base provides a template that predefines common dimensions of information such as intelligence, emotional intelligence, and interests, to facilitate the prioritized filling of these attributes when creating new instances (Booch, 2007).

The basic attributes of an entity include ID (index), Name (identifier), and Description (brief description). This design allows game masters to quickly grasp an overview of an entity without needing to access its complete detailed information, thereby facilitating the assessment of the entity's relevance to the upcoming storyline.

Variables constructed for game logic and judgment are broadly divided into two categories: int type for direct mathematical operations, and string type for handling logical analysis that requires language model support. For example, the population of a city can be intuitively represented and involved in calculations using an int type, while cultural characteristics need to be described using string type to portray its history and beliefs (information that cannot be directly quantified).

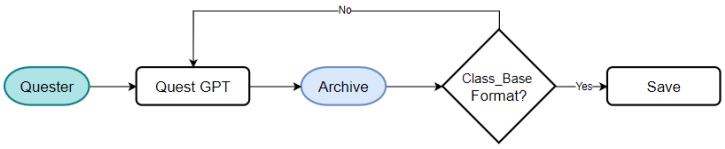
In the design of Class_Base, ID, Name, and Description are first set as the basic variables, followed by the introduction of Dictionary_Int and Dictionary_String, allowing subclasses to customize more dimensions of variables as needed.

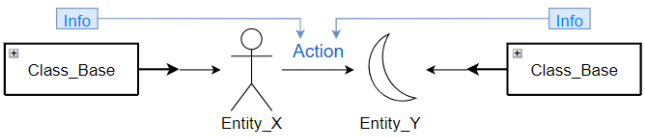


The game asset system based on Class_Base has the following advantages: i) Standardized information: Each entity has standardized quantified information, facilitating the understanding of the game world and objects by LLMs and players. For instance, information about the subject and object in action is used to generate actions.

ii) Flexibility: LLMs can design new Class templates to support new categories of entities, thereby enriching the composition of the game world.

iii) Stability: Based on Class_Base JSON format files, information can be securely invoked during game computations. As shown in the diagram, when using entity information generated by LLMs, Class_Base can first be used as a standard for format verification to ensure the safety and applicability of information. If the information does not meet the standards, the system can promptly provide feedback on errors and require LLMs to regenerate safe information.

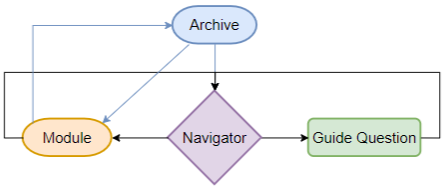




2. Independent Agent Module: Sharing Responsibilities with the Game Master

In traditional DnD games, the role of the Dungeon Master (DM) is crucial, as they are responsible for maintaining the coherence of the game's background and responding to player actions (Zhu, 2023). However, if all functions are handled by a single LLM (Large Language Model), it would lead to rapid resource depletion and could potentially impact the quality of task handling. By implementing a multi-agent system, focused functions and clear division of labor can be achieved (Talebirad & Nadiri, 2023; Park et al.), thereby enhancing efficiency and response speed.

As illustrated in the diagram, within the GPT-World system. The main LLM—Navigator—is responsible for guiding players to engage in the game process. Based on the needs of the process, it then calls the appropriate functional modules. Besides issuing commands to invoke modules, the Navigator can also simply ask players about their intentions to determine the next process requirements. When issuing commands to modules, the Navigator needs to provide the index of the related Entity. For example, when instructing the module Talker to converse with a player, it needs to specify detailed information about the Chatbot's role, including abilities and personality. Therefore, Navigator and Modules have frequent information sharing with the Archive, based on a secure JSON format. By reading information, it supports the LLM in assessing the game environment and providing information to players. Through writing information, new scripts and Entities are generated, or based on the actions and choices of players, the game environment is rewritten.

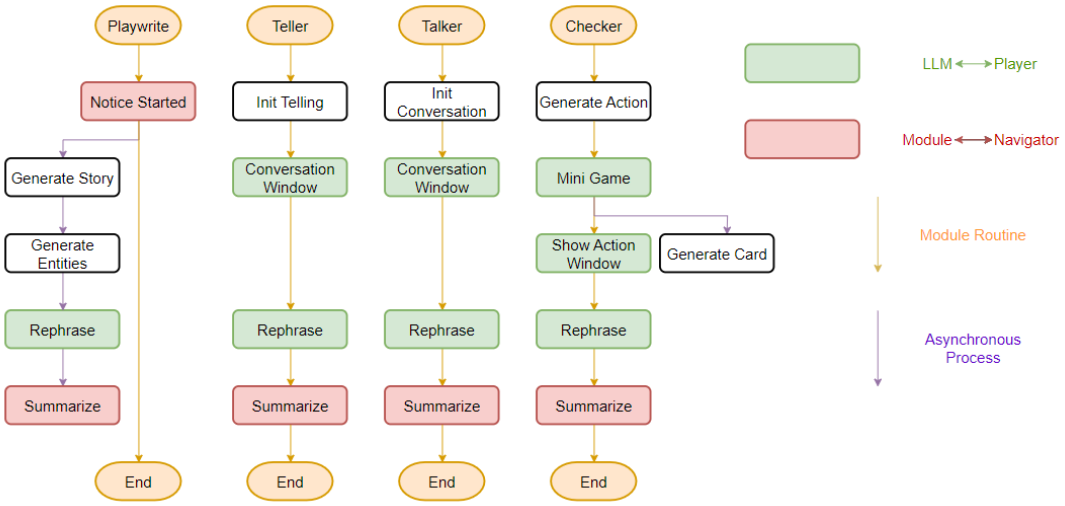


GPT-World currently supports modules such as Playwright, Teller, Talker, and Check. Playwright generates scripts based on plots required by Navigator, then creates information on entities such as characters and locations needed for the performance of the script. Teller narrates scenes and encounters to the player in a dialogic manner, draws pictures of the scene, helping the player understand the current situation. As a chatbot, Talker engages in dialogue with NPCs and players, providing information and fulfilling the players' needs for companionship and social interaction within the game. After the conversation, it also modifies the relationship between the player and NPCs based on the content of the dialogue.

Checker generates detailed descriptions of actions when players make choices or are about to act, assessing the difficulty of the action based on specific information about the subject and object involved, and then uses this difficulty as a threshold for determining the success of dice rolls. Check also categorizes actions with labels. During checks, actions can receive bonus points from skill cards of the same category, making it easier for players to succeed in checks. If an action is successful, it updates

the information of the subject and object involved, allowing the player's choices and actions to tangibly change the game environment. Successful actions also generate skill cards, benefiting the next check of the same type.

As shown in the diagram, the green frames indicate the modules' functions that directly interact with players in the game interface, such as Teller and Talker engaging in conversations or generating images in the Conversation Window. During the Check module process, players can roll dice and view action information in the action information window. The red frames indicate modules communicating with Navigator, reporting progress or summarizing the module's operations and results. Tasks for generating new plots and entities are executed asynchronously to avoid delays caused by asset generation.



Each module has a standardized structure and functions, including a Rephase phase to summarize experiences and a Summarize phase to report progress to Navigator. Additionally, each module can independently terminate its process. For example, in the Talker module, if an NPC no longer wishes to communicate with the player or knows the conversation has ended, it will automatically stop the module. This reduces player control and allows the module to transition more naturally and smoothly to Navigator. This modular design not only reduces the load on the main LLM but also enhances the system's stability and scalability.

Below is the prompt that requires Navigator to send instructions to the module.

```
"You are the Manager of the DnD game and are responsible for controlling the flow of the game and maintaining the rules of the game. You can only perform two types of actions: " +
"1. Directly Ask questions to player to guide their exploration, without json format " +
"2. Convey other tasks such as descriptions, generation, checks, etc., to subordinates through JSON commands. \r\n" +
"The structure of the instruction should include action type, detailed description, subject, and optional object information. The following are various Scenario JSON templates: \r\n" +

//Playwright
"\r\nTo design new challenge plot:" +
"{\"actionType\": \"CreateChallenge\", \"description\": \"Describe the content and expected effects of the challenge\"}" +
"\r\nTo generate new entity:" +
"{\"actionType\": \"CreateEntity\", \"description\": \"Describe the newly generated entity's features\", \"entityType\": \"Entity type\", \"entityName\": \"Name\"}" +
//Teller
"\r\nTo describe the environment the player is in, or the events that occur:" +
"{\"actionType\": \"DescribeSituation\", \"description\": \"Instructions given to the GM, what he needs to describe to the player\"}" +
//Talker
"\r\nTo start conversation with npc chatbot:" +
"{\"actionType\": \"InitiateConversation\", \"description\": \"Describe the content and purpose of the conversation\", \"entityID\": \"find conversation target ID from Archive, should not be player ID\"}" +
//Checker
"\r\nTo perform a capability check (object field is optional):" +
"{\"actionType\": \"ActionCheck\", \"description\": \"Describe the specific action and the capability checks involved\", \"subject\": {\"entityType\": \"Class type\", \"entityID\": \"Character ID (int), usually for player\"}, " +
"\"object\": {\"entityType\": \"Object class type\", \"entityID\": \"Object ID\"}}" +
"or: "+ "{\"actionType\": \"ActionCheck\", \"description\": \"Describe the specific action and the capability checks involved\", \"subject\": {\"entityType\": \"Class type\", \"entityID\": \"Character ID (int), usually for player\"}}" +


"\r\nNote:\r\n " +
"Please use {} for the JSON format; do not substitute with **. Each variable in the JSON format must be provided. \r\n" +
"Optional fields may be omitted unless required.\r\n " +
"Make sure all entityTypes and entityIDs are as specified in the game documentation to avoid confusion and errors.\r\n " +
"Providing adequate descriptions can help subordinates better understand the purpose and expected results of each action.\r\n " +
"Each submitted instruction should contain only one operation to keep it clear and orderly.\r\n\r\n";
```

3. Json Asset Reading and Writing System and Module Division of Labor: Supporting Long-Term, Stable AI DnD Operation.

To support a long-term AI DnD gaming experience, there must be an Archive system capable of freely reading and writing to manage and maintain a large volume of game information. This is crucial for maintaining the coherence and depth of the game, especially in AI DnD sessions that last from months to years, where the world created by players and the DM needs to be based on common sense reasoning and thematic consistency (Ammanabrolu, 2020; Martin, Sood, & Riedl, 2018; Zhou et al., 2023).

In the system, the Navigator acts as the central control hub with access to brief information about Entities in the Archive, while specific functional modules can access detailed information as required by their tasks, ensuring decision-making accuracy and information consistency. This strategy of accessing information as needed reduces resource wastage and allows each module to focus more on its respective tasks, enhancing processing efficiency and the coherence of the gaming experience.

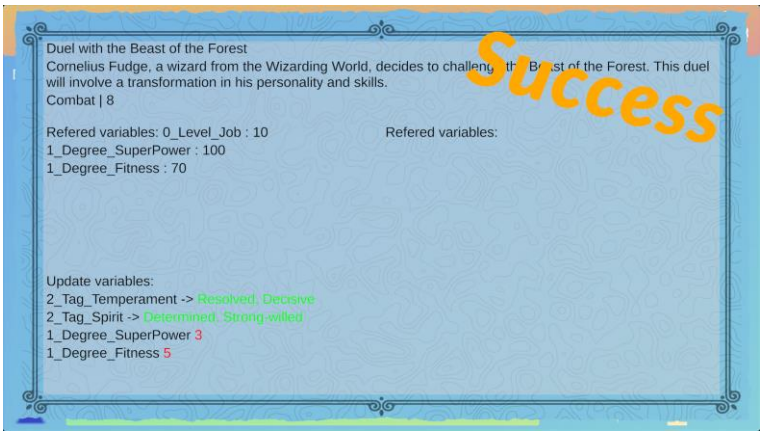
In summary, the Json-based asset reading and writing system and the modular agent design not only enhance the flexibility and stability of the game but also provide strong technical support for supporting complex, long-duration AI DnD games.

Part 2. Quantitative Entity Information for More Reliable AI Logic and Support for Gameplay Involving Mathematical Operations

Current AI interactive narrative games, such as AI Dungeon (Latitude 2020), often lack a convincing logical foundation and effective player control. In these games, when a large language model (LLM) needs to assess the difficulty of an action, it typically relies only on a brief description of the action's agent. This approach lacks detailed and quantified information, leading to often unreasonable difficulty assessments.

To address this issue, this project adopts the Chain-of-Thought method (Wei et al. 2022), which simulates human reasoning processes to guide the model in gradually deriving outcomes. As shown in the diagram, when generating the difficulty of an action, we consider the quantified information of the action's agent. This method

allows for multi-dimensional referencing, thus making the difficulty assessments more reasonable.



The prompt for generating the action is as follows:

```
"Referring to the variables in the two Entities, and Description. Describe an task that the Entity A does in the RPG game, and set the task difficulty."+"\r\n" +
 "1. set Level_Difficulty, on a scale of 1-10. Based on values of Entity A and Entity B(if has), evaluate the task difficulty."+"\r\n" +
 "2. fullfill other varaibles" +"\r\n" +

"The results are stored as Json file in the structure of Class_Action" +
"Please only return the Json file. Any additional explanations or symbols such as ``` will cause errors"+"\r\n" +

Resources.Load<TextAsset>("Class_Action").text + "\r\n"
```

To generate stable Json responses and provide clear descriptions of variables, the project has verified that first providing descriptions of the Class variables followed by a Json template achieves stable and excellent results. The specific implementation method is shown by loading the description of Class variables from the code Resources.Load<TextAsset>("Class_Action").text.

```
public class Class_Action
{
    public int ID = -1;
    public string Name = "";
    public string Description = "";

    //Select within enum Type_Action (Harvest, Produce,  Invent, Buff, Combat, Military,  Commerce, Communicate, Manage, Discover, Stealth)
    public string Tag_ActionType = "";
    //Evaluate the task difficulty on a scale from 1 to 10, Based on EntityA's capabilities and the difficulty of the task.
    public int Level_Difficulty= -1;

    ///ClassA is the class of the Entity A, and ClassB is the class of the Entity B. The variable names must be exactly the same.
    /// <summary>
    /// Variables related to the success of the action. The value of the variable that was originally an int is also written as a string.
    /// The first string is the variable name and the second string is the value of the variable.
    /// </summary>
    public Dictionary<String, String> ClassAReferredDic;
    public Dictionary<String, String> ClassBReferredDic;

    /// <summary>
    /// Variables that will be affected after the action is successful. The value of the variable that was originally an int is also written as a string.
    /// The first string is the variable name
    /// the second string:
    /// (originally int) is the value to add(can be negative), the amount of change
    /// (originally string) is the new description overridden, the result after the change
    /// </summary>
    public Dictionary<String, String> ClassAChangedDic;
    public Dictionary<String, String> ClassBChangedDic;
}
```

```
//Json template:
  {
    "ID": "int32",
    "Name": "string",
    "Description": "string",
    "Level_Difficulty ": "int32",
    "Tag_ActionType": "string",


    "ClassAReferredDic;": {
      "variableNameA1": "string",
      "variableNameA2": "string"
    },
    "ClassBReferredDic": {
      "variableNameB1": "string",
      "variableNameB2": "string"
    },
    "ClassAChangedDic": {
      "variableNameA1": "string",
      "variableNameA2": "string"
    },
    "ClassBChangedDic": {
      "variableNameB1": "string",
      "variableNameB2": "string"
    }
  }
```

After an action is completed, the variables of the involved entities can be changed based on the results of the action, allowing the player's actions to substantively alter the game world on a quantified and reasonable basis. This method of judgment and change, which includes reasonable quantitative dimensions and values, compared to the vague descriptions and over-reliance on brief textual information in AI Dungeon, supports more enduring and accumulative player actions.

More importantly, for future AI-Native Games, it is necessary to combine AI's creativity and flexibility with the gameplay calculation advantages based on mechanics found in traditional video games. For instance, "Baldur's Gate 3" (Larian Studios, 2023), a game built on the DnD's D20 system, features a complex and robust combat system, which is currently lacking in AI games that rely on LLM for text processing. By quantifying the variables of entities in the game in multiple dimensions to support mathematical operations, it allows for the adoption of gameplay calculations based on mathematics found in traditional video games.

Project Milestones

First Iteration (until 2023/11/22):

The initiation of quantitative functionality verification for AI-Native Game.

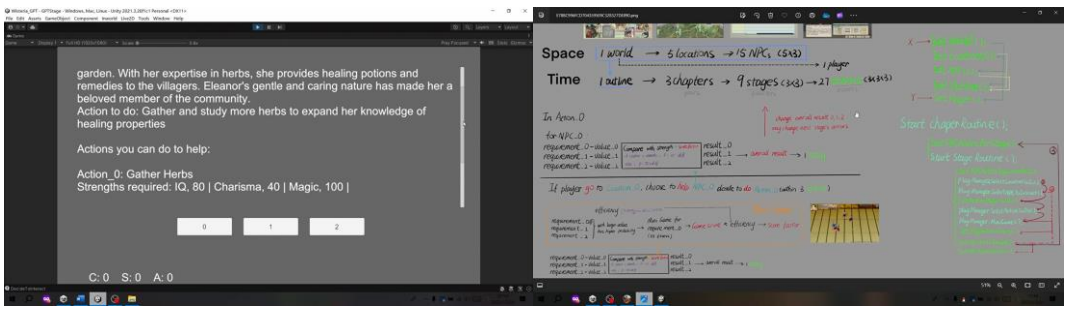
The content of the game features a fish by the lakeside of Wisteria who loves dreaming. In her heart, she is a girl fond of magic. GPT-3.5 predicts her dream content based on her current needs using Maslow's hierarchy of needs. Additionally, players can converse with her, enabled by the Inworld plugin (Inworld AI, 2024), and the 2D cartoon animations are created using Live2d Cubism (Live2D Inc, 2012).

Second Iteration (until 2023/12/22):

Initial setup for JSON secure writing to the Archive was laid out, with attempts to build a more complex game system utilizing GPT-3.5.

The goal at this time was to create an AI-driven "Taiko Risshiden V" (Koei Tecmo Games, 2022). The AI could drive NPCs to choose actions beneficial to themselves based on their own information.

At this point, the AI was able to generate the necessary game entities based on the world view, but the framework was rigid, with each class having its own format and required functions. The final outcome allowed players to choose an NPC they wanted to help, and then the AI would suggest several actions that the player could take.



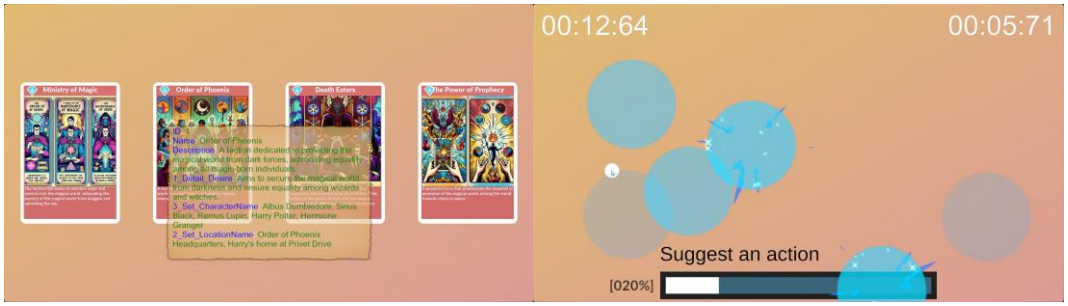
Third Iteration (until 2024/3/2):

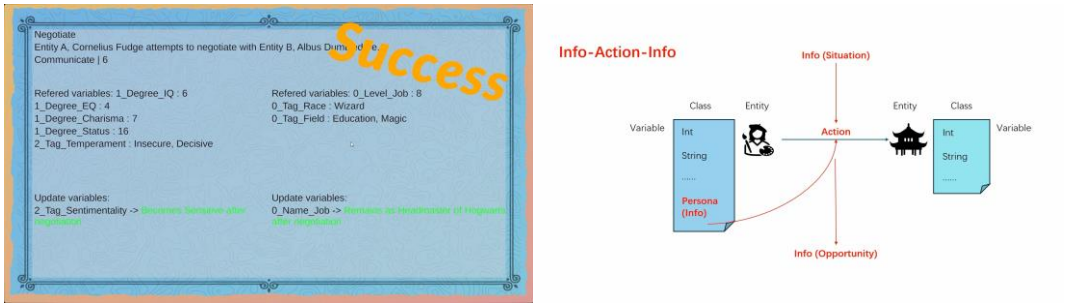
Drawing from the design philosophy of "Chris Crawford On Interactive Storytelling" (Crawford, 2012), a clear direction was set for an action and information-driven design.

The intended AI recreation of "Taiko 5" was a complex and massive game. During the winter break and the Spring Festival, extensive game design was done, contemplating how to enable AI to drive an interesting game experience, though there was no time for implementation.

Using Class_Base as a parent to generate entities under each class, and related functions became more generic, making the framework more streamlined and flexible.

GPT-3.5 was replaced with GPT-4, which allowed for smarter judgment by AI, enabling a more flexible and natural integration of the game framework. Entity Images were created using Dalle3, and the game process from the second iteration was beautified using the Feel plugin(More Mountains, 2024).

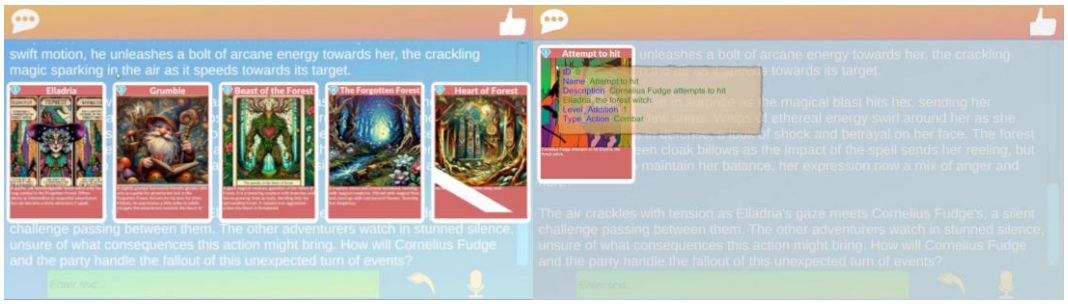




Fourth Iteration (until 2024/3/29):

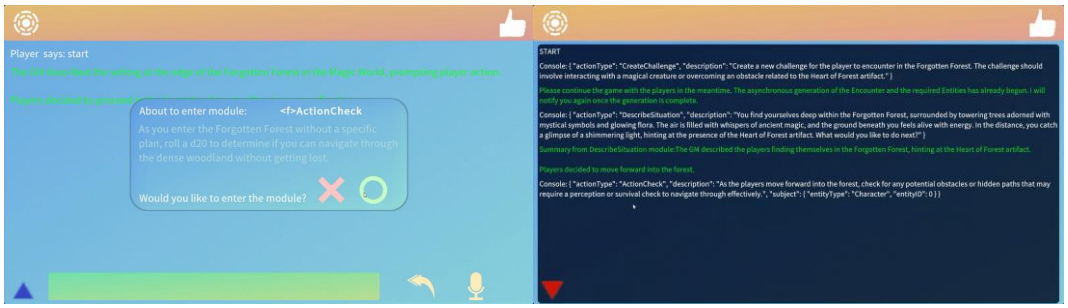
The decision was made to shift from AI's "Taiko Risshiden V" to AI-driven DnD. The game supports functions similar to AI Dungeon: a single LLM-driven Game Master guides players through a text-based adventure game. Unlike AI Dungeon, it supports reading and writing multi-dimensional quantified information of AI-generated entities. The chatbot is implemented by an independent GPT-4 agent, allowing players to choose which NPC they wish to converse with. A new gameplay element was added where successful player actions generate cards that are beneficial for subsequent checks.

This iteration utilized GPT-4 Tools to enable the Game Master to automatically switch between Check and Conversation game states.



Fifth Iteration (until 2024/4/25):

The most recent iteration has mainly focused on academic research and paper preparation. By introducing the Navigator to manage subordinate modules, system flexibility and efficiency were further enhanced. Improvements were made to the system to stabilize and sharpen state transitions, and techniques such as asynchronous loading were employed to reduce response times, significantly enhancing game fluidity.



Takeaways and Future Works

The project was conducted concurrently on academic and industrial tracks. Throughout the process, I received considerable help from my seniors. Professor Duh consistently challenged me with the question "Why," emphasizing that as a graduate student, my solutions should not be baseless ideas but should be backed by evidence. Why is the problem important? Why should it be approached this way? Why is this approach better? Mr. Trevor Chan frequently asked me "When" questions about project readiness, such as when the Pitch Deck would be ready, when the business research and cost budgeting would be done, and when the game would be released. Meanwhile, I constantly asked myself "How" questions, such as how to implement a system based on quantification and automated read-write through LLM? How to flexibly integrate it into the game? How to make the gaming experience more engaging?

Professor Duh's high standards forced me to think and construct methods at a higher level, leading me to gradually make the framework more universal and flexible, with new designs like Class_Base and modularization, significantly restructuring the system twice. Mr. Chan's attention and encouragement were major motivators for me to persist with the project in its early stages, making me take each timeline he inquired about seriously and plan accordingly.

During the project, I also conducted some validation work. At the end of each iteration cycle, I would get feedback from game design peers to optimize the next iteration. After the fourth iteration, I received comments from Reddit; some suggested I turn to MUD games, others hoped I would strengthen the story's coherence and stable information storage and tracking of game assets. During the fifth iteration, I exchanged ideas with Ilya Berlikin, who was also working on an AI DnD product, and realized that managing game flow with an LLM was indeed smoother. This urged me to ensure precise and natural integration between modules and the Navigator. However, I also recognized the strengths of my game: supporting complex gameplay, an independent Chatbot, clearer judgment logic after quantification, and long-term maintenance of game assets and information. These aspects meant that my framework under an AI Native Game could better satisfy the conditions for modern electronic games that keep players engaged for longer periods.

In the academic aspect, the project still lacks solid validation of the argument, which requires finding a test group to verify the actual effects of the quantification and automated read-write system. In the industrial aspect, the game still needs more refinement. When checking actions, replacing dice rolls with various mini-games could be considered. For future work, I plan to add more visual elements, such as supporting customized and smarter 2D art production workflows through ComfyUI, implementing a 2D sandbox game map, adding more functional modules, and refining and optimizing the gaming experience.

Beyond academic and industrial exploration, I plan to attempt some commercial

arrangements, such as releasing the game on Steam or Itch.io to seek user experience feedback and make preliminary attempts at commercial actions based on AI products. Considering that mass-oriented products require a stable and mature experience, current AI technologies may not be perfect. I may initially open-source the project on GitHub, hoping to continue refining and exploring AI and gaming with like-minded developers, and even future interaction methods.

Conclusion

The GTP-World project focuses on addressing inconsistencies in game contexts and unclear logic, enhancing the complex mechanism handling and long-term operability of "AI-Native" games through quantification and automated read-write. I hope this project highlights the standardization of information brought by quantification and its positive impact on the integration of complex game mechanisms, as well as the contributions of AI's autonomous read-write to content consistency and long-term stable operation. Despite challenges, as Schell (2020) stated, "We game designers must unite," I hope this work opens new possibilities for the future of AI-native games.

Reference List

Ammanabrolu, P., & Riedl, M. (2019). *Playing text-adventure games with graph-based deep reinforcement learning*. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 3557–3565).

Ammanabrolu, P., Cheung, W., Tu, D., Broniec, W., & Riedl, M. (2020). *Bringing stories alive: Generating interactive fiction worlds*. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 16(1), 3-9.

Bergström, K., Jonsson, S., & Björk, S. (2010). *Undercurrents: A computer-based gameplay tool to support tabletop roleplaying*. In Proceedings of the 2010 International DiGRA Nordic Conference: Experiencing Games: Games, Play, and Players. Retrieved from http://www.digra.org/wp-content/uploads/digital-library/10343.50118.pdf

Ceze, L. (2022). *Don't be fooled by AI washing: 3 questions to ask before you invest*. Retrieved July 8, 2023, from https://venturebeat.com/datadecisionmakers/dont-be-fooled-by-ai-washing-3-questions-to-ask-before-you-invest/

Crawford, C. (2012). *Chris Crawford on Interactive Storytelling* (2nd ed.). Pearson Education, Limited.

D&D Beyond. (2017). Retrieved from https://www.dndbeyond.com/.

D&D Beyond. (2022). *Avrae*. Retrieved from https://avrae.io/

Ellis, S., & Hendler, J. (2017). *Computers play chess, computers play go...Humans play Dungeons & Dragons*. IEEE Intelligent Systems, 32(4), 31-34.

Friends & Fables. (n.d.). Retrieved from https://www.fables.gg/

Gallotta, R., Todd, G., Zammit, M., Earle, S., Liapis, A., Togelius, J., & Yannakakis, G. N. (2024). *Large language models and games: A survey and roadmap*. arXiv preprint arXiv:2402.18659.

Gwertzman, J., & Soslow, J. (2022, November 17). *The generative AI revolution in games*. Andreessen Horowitz. Retrieved April 16, 2024, from https://a16z.com/the-generative-ai-revolution-in-games/

Hidden Door. (2024). Retrieved from https://www.hiddendoor.co/

Inworld AI. (2024). *Inworld: AI-powered gameplay*. Retrieved from https://inworld.ai/

Koei Tecmo Games. (2022). *Taiko Risshiden V DX official site*. Retrieved from https://www.gamecity.ne.jp/taikou5dx/

Studios. (2023). *Baldur's Gate III*. Retrieved from https://baldursgate3.game/

Latitude. (2020). *AI Dungeon*. Retrieved July 8, 2023, from https://play.aidungeon.io/main/home

Live2D Inc. (2012). *Live2D Cubism* [Software]. Retrieved from https://www.live2d.com

Louis, A., & Sutton, C. (2018). *Deep Dungeons and Dragons: Learning character-action interactions from role-playing game transcripts*. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), 708-713.

Martin, L.J., Sood, S., & Riedl, M.O. (2018). *Dungeons and DQNs: Toward Reinforcement Learning Agents that Play Tabletop Roleplaying Games*. INT/WICED@AIIDE.

Muller, M., Chilton, L. B., Kantosalo, A., Martin, C. P., & Walsh, G. (2022). *GenAICHI: Generative AI and HCI*. In CHI Conference on Human Factors in Computing Systems Extended Abstracts (pp. 1-7).

OpenAI. (2019, November 5). *GPT-2: 1.5B release*. Retrieved from https://openai.com/research/gpt-2-1-5b-release

OPENAI. (2023b). *GPT-4*. Retrieved July 8, 2023, from https://openai.com/research/gpt-4

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). *Generative agents: Interactive simulacra of human behavior*. In Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, 1-22.

Salen, K., & Zimmerman, E. (2003). *Rules of play: Game design fundamentals*. MIT Press.

Samuel, B., Treanor, M., & McCoy, J. (2021). *Design considerations for creating AI-based gameplay*. In International Conference on Interactive Digital Storytelling (pp. 123-134). Springer.

Santiago, J. M., III, Parayno, R. L., Deja, J. A., & Samson, B. P. V. (2023). *Rolling the Dice: Imagining generative AI as a Dungeons & Dragons storytelling companion*. arXiv:2304.01860 [cs]. Retrieved from https://arxiv.org/abs/2304.01860

Schell, J. (2020). *The art of game design: A book of lenses* (3rd ed.). CRC Press.

Scriven, P. (2021). *From tabletop to screen: Playing Dungeons and Dragons during COVID-19*. Societies, 11(4), 1-13. https://doi.org/10.3390/soc11040125

Sidhu, P., & Carter, M. (2024). *Memorable play in Dungeons & Dragons: Understanding the relationship between TTRPG design and peak play experiences*. Proceedings of the 57th Hawaii International Conference on System Sciences. Retrieved from https://hdl.handle.net/10125/106707

Sudhakaran, S., González-Duque, M., Glanois, C., Freiberger, M., Najarro, E., & Risi, S. (2023). *MarioGPT: Open-ended text2level generation through large language models*. arXiv preprint arXiv:2302.05981. Retrieved from https://arxiv.org/abs/2302.05981

Sun, Y., Li, Z., Fang, K., Lee, C. H., & Asadipour, A. (2023). *Language as reality: A co-creative storytelling game experience in 1001 nights using generative AI*. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 19(1), 425-434. https://doi.org/10.1609/aiide.v19i1.27539
Talebirad, Y., & Nadiri, A. (2023). *Multi-agent collaboration: Harnessing the power of intelligent LLM agents*. arXiv preprint arXiv:2306.03314.

Van Stegeren, J., & Myśliwiec, J. (2021). *Fine-tuning GPT-2 on annotated RPG quests for NPC dialogue generation*. In Proceedings of the 16th International Conference on the Foundations of Digital Games (FDG '21) (Article 2, pp. 1–8). Association for Computing Machinery. https://doi.org/10.1145/3472538.3472595

Värtinen, S., Hämäläinen, P., & Guckelsberger, C. (2024). *Generating role-playing game quests with GPT language models*. IEEE Transactions on Games, 16, 127-139.

Yan, V. (2023). *Yandere AI girlfriend simulator*. Retrieved July 8, 2023, from https://helixngc7293.itch.io/yandere-ai-girlfriend-simulator

Yannakakis, G. N., & Togelius, J. (2014). *A panorama of artificial and computational intelligence in games*. IEEE Transactions on Computational Intelligence and AI in Games, 7(4), 317-335.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2022). *React: Synergizing reasoning and acting in language models*. arXiv preprint arXiv:2210.03629.

Zhou, P., Zhu, A., Hu, J., Pujara, J., Ren, X., Callison-Burch, C., Choi, Y., & Ammanabrolu, P. (2023). *I Cast Detect Thoughts: Learning to converse and guide with intents and theory-of-mind in Dungeons and Dragons*. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 11136–11155). Toronto, Canada: Association for Computational Linguistics.

Zhu, A., Martin, L., Head, A., & Callison-Burch, C. (2023). *CALYPSO: LLMs as dungeon masters' assistants*. In Proceedings of the Nineteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (pp. 380-390). AAAI Press. https://doi.org/10.1609/aiide.v19i1.27534