



Towards Authoring Open-Ended Behaviors for Narrative Puzzle Games with Large Language Model Support

Britney Ngaw
New York University
New York, New York, USA

João Sedoc
New York University
New York, New York, USA

Grishma Jena
IBM
San Francisco, California, USA

Aline Normoyle
Bryn Mawr College
Bryn Mawr, Pennsylvania, USA



Figure 1: Screenshots from our Starship Voyager Narrative Game. Players use natural language text to progress through the story. Left, the game starts in the near future in orbit around Mars. Center, the player is a microbiologist on board an orbital shuttle where they can speak with their colleagues Lieutenant John Kelly and Kumagawa. Right, in the far future, the player is a Starfleet officer trapped in their crew quarters and tasked with finding a way out.

ABSTRACT

Designing games with branching story lines, object annotations, scene details, and dialog can be challenging due to the intensive authoring required. We investigate the potential for authoring open-ended behaviors for point-and-click narrative games using GPT-3.5, a large language model. In our approach, we extend a behavior tree scripting system with nodes that query GPT-3.5 to generate object descriptions, conversations with characters, and responses to player actions. GPT-3.5 is used to generate content when it hasn't been scripted manually and to update game state by asking questions about whether a player's input achieves a particular game goal. We demonstrate our approach with puzzles based on scenes from an episode of Star Trek Voyager. Our approach aims to blend a specific plot with open-ended story elements while keeping the authoring work minimal. Based on a pilot study of 16 participants and our own testing, we find that the generated responses have high coherency and show signs of humor and novelty, but that utterances could be improved to be more interesting and better support the designer's intent.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

FDG 2024, May 21–24, 2024, Worcester, MA, USA

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0955-5/24/05
<https://doi.org/10.1145/3649921.3656975>

CCS CONCEPTS

- Software and its engineering → Interactive games;
- Information systems → Language models.

KEYWORDS

Large Language Models, Behavior Trees, Game Design, Game Authoring Tools, Procedural Narrative Generation,

ACM Reference Format:

Britney Ngaw, Grishma Jena, João Sedoc, and Aline Normoyle. 2024. Towards Authoring Open-Ended Behaviors for Narrative Puzzle Games with Large Language Model Support. In *Proceedings of the 19th International Conference on the Foundations of Digital Games (FDG 2024), May 21–24, 2024, Worcester, MA, USA*. ACM, New York, NY, USA, 4 pages. <https://doi.org/10.1145/3649921.3656975>

1 INTRODUCTION

Plot-centric narrative games, such as the *Zork* and *The Wolf Among Us* combine elements of exploration, puzzle solving, and interactions with characters to lead the player through a story. These games typically integrate puzzles into the environment. By solving puzzles, players progress through the story. These games can be challenging to design because they require a large amount of authoring to generate branching story lines, annotate objects and their affordances, specify scene details, and author dialog. Furthermore, to give the player a sense of agency, the designer needs to anticipate potential actions and implement their effects on the environment and story.

In this work, we investigate the potential for large language models (LLM), such as GPT-3.5 [8], to augment manually authored game elements with generated content. Towards this goal, we prototype a behavior tree based scripting system that contains nodes with integrated GPT-3.5 features. The puzzle designer crafts the experience using a text-based script that allows them to specify GPT-3.5 prompts, trigger animations and sounds, set puzzle goals, list the scene progression, and explicitly set up important dialog. The resulting script is similar to a template for the puzzle. However, rather than a database to fill in the details of the template, GPT-3.5 fills in details on the fly. Thus, the LLM can be used to fill in gaps where the designer has not written content, to add variety to the game environment, or to serve as placeholders for content that will be fleshed out later.

As a proof of concept, we designed two puzzles based on an episode of Star Trek Voyager. We choose Star Trek as a game premise because of its wide-spread appeal, its wide variety of self-contained episodes with well-crafted plot structures, and its extensive online transcripts and fan fiction communities that make it well-supported by GPT-3.5. Our game script is based on an episode transcript from which we pull major plot points and dialog. We use the original dialog and plot points as content and objectives for our puzzles. GPT-3.5 generates content for situations we don't explicitly handle. Overall, the approach has promise for giving varied content that is appropriate for the scene, character, and puzzle aims. Based on a small pilot study of 16 participants, players generally liked conversing with our characters but found our game mechanic based on open-ended actions confusing. We also analyze the dialog transcripts from our pilot study and our own testing. From 17 transcripts containing 160 LLM generated utterances, we identified four mistakes involving wrong information, repetition, contradiction, and breaking the fourth wall. These results help guide future work into how we can best leverage LLMs to augment human authored narrative games.

2 RELATED WORK

Many researchers are investigating the use of large-language models for games [4] and social simulation [9]. For example, Park *et al.*[9] used Chat-GPT as a generative backend to simulate a town consisting of 25 agents. Kumaran *et. al.* [6] used LLMs to generate 3D scenes based on a user's description. Unlike these systems, in which a large portion of the narrative is generated, this work investigates the use of LLMs for filling in story details, as specified by the game designer. Thus, the LLM acts as an assistant Game Moderator who goal is to be a partner in co-creation[7].

Our approach is similar to other template-based procedural content generation methods, in which the game details are filled using a database or corpus of text. For example, in an early work aimed at point-and-click adventure games, Fernández-Vara and Thomson [2] described their puzzle-dice system, which allowed designers to specify hierarchical structures of puzzles at a high-level. The specifics of each puzzle were filled in based on a database of objects, affordances, and states. Similarly, Dart *et al.* [1] used a database to create variety in puzzles and Swanson and Gordon's *Say Anything*[10] used a corpus of blogs to co-create stories between a player and the system. LLMs now give us an opportunity to query knowledge

– such as whether a chair can prop open a door – rather than rely on existing databases of objects.

3 GAME DESIGN

Our proof of concept implements two puzzles based on episode eight of season six from *Star Trek: Voyager*, entitled *One Small Step* [11]. We chose this episode because it is highly rated and provides a good example of a well-structured, self-contained plot. The first scene introduces the game interface and asks the player to talk with two non-player characters, commanding officer, Lieutenant John Kelly, and Mars scientist, Kumagawa. The second scene presents a puzzle to the player where they must enter actions via text to try to escape their crew quarters, for example, by fixing the door, propping the door open, or calling for help.

3.1 Prompts

We query GPT-3.5 using an HTTPS connection to OpenAI's REST API. Our current implementation makes two types of requests to OpenAI: a *completion* request and an *embedding* request. *Completion* requests generate text responses. *Embedding* requests return feature vectors for text that we use to test similarity to given phrases using cosine similarity. A cosine similarity close to 1.0 indicates two very similar phrases. A cosine similarity close to 0 indicates two unrelated phrases.

3.2 Scenario Authoring

The authoring system is based on a behavior tree that supports nodes for configuring the story, interacting with the player, and showing feedback via visuals, animations, and sounds. The system is built using Unity. Authoring requires setting up scene assets and then configuring behaviors that connect these assets to user input and prompts to GPT-3.5. Figure 2 shows an overview. Behaviors are loaded from a script when the level loads, along with pre-configured information about the characters in the scene, such as their personalities, previous knowledge, and personal history. When the player asks a character about themselves, this information helps GPT-3.5 answer in a consistent and believable way. With our system, the root behavior node is always a sequence node. Each line in the script corresponds to a behavior. Control nodes – such as loops, select, and conditionals – have children nodes. Within the game system, behaviors implement logic. Any persistent game state is stored in a World object that is shared by all behavior nodes. Conceptually, the behavior tree implements a story manager that guides the player through the plot. With this setup, we define story, character, dialog, and action nodes that support experimenting with different prompts to create puzzles.

3.2.1 Story and Character Nodes. Our behavior tree implementation supports custom nodes for configuring story context and character details that are used later when we query GPT-3.5 for character dialog and responses to player actions. *InitCharacter* loads three text files, containing a character's personality, history, and knowledge. *SetState:PromptStyle* sets a style that we want GPT-3.5 to use. *Story* sets scene context. *Description* queries GPT-3.5 for a description of an object in the scene. Below is an excerpt from an input script that initializes a scene. Each line corresponds to a node that is executed in sequence.

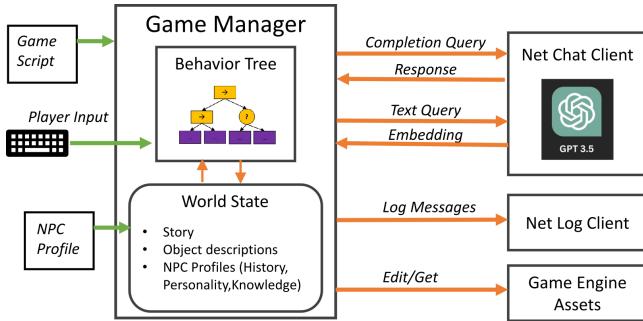


Figure 2: System Architecture.

```

InitCharacter: KELLY
InitCharacter: KUMAGAWA
SetState: PromptStyle, In a short sentence with a sense
of humor,
Story: Kumagawa is drilling on the Mars Surface to get
soil samples.
Story: The player is a young microbiologist
Story: The player is sitting next to John Kelly.
Description: Photo, prompt: Describe a sentimental photo.
  
```

3.2.2 Dialog Nodes. Our behavior tree implementation supports both hard-coded and open-ended dialog. Hard-coded dialog is implemented with a *Say* behavior that shows a box with text next to the character. *Say* behaviors are configured from our input script using the name of the character, such as KELLY for Lieutenant Kelly. Open-ended dialog is implemented using a *Dialog* behavior where the player enters text and then the character shows their response. Below is an excerpt that shows how these behaviors are scripted. *WaitClick* corresponds to a conditional node. If the user clicks on Lieutenant Kelly, we run the two behaviors beneath it in sequence.

```

WaitClick: KELLY
  KELLY: Hey Rookie, are you looking forward to getting
  new soil samples?
  Dialog: KELLY, Lieutenant John Kelly looks at you
  expectantly.
End
  
```

The *Dialog* node creates and sends a prompt to GPT-3.5 with the following form. [NPC] is the character's name. [Personality] and [Knowledge] are initialized in *InitCharacter*. [Story List] and [Dialog History] contain text from *Story* nodes as well as dialog history.

```

[NPC] has these character traits: [Personality].
[NPC] knows about these things: [Knowledge].
The story so far: [Story List].
This is the dialog so far: [Dialog History].
The player says to [NPC]: [Player Input].
Respond in one sentence as if you were [NPC].
  
```

3.2.3 Action Nodes. Our behavior tree implementation supports open-ended actions from the player using a compound behavior. This behavior implements several conditional nodes that implement a chain of prompts. Below is an example script that configures an action.

```

GetAction: Describe an action you want to take.
IfNotValidAction:
  Inform: Interesting idea, but you don't have
  
```

```

the tools or abilities to perform this action.
End
IfCompare: 0.9, prop open the door
  Inform: The door slams into the makeshift doorstop
but leaves a space for you to fit through.
End
End
  
```

GetAction implements a behavior that asks the player for a choice and then gives feedback. This behavior contains any number of conditional nodes that give feedback on the user's choice. Above, *IfNotValidAction* checks whether the action is valid, based on the traits of the player character and the objects in the scene. *IfCompare* uses Open AI's embedding API to check the similarity of the user's action with the phrase "prop open the door". If these checks pass, we do two last prompts to execute the action: one to ask whether the action solves any problems for the player. Successful actions are automatically added to the story context.

GetAction makes multiple queries to GPT-3.5. *IfNotValidAction* uses the following prompt to ask whether the player can perform a proposed action.

```

This is the story so far: [Story List]
The room only contains the following items: [Object List]
Is it possible for the player to do the following action
using only the items in the room or their own
capabilities? [Player Input]
Answer yes or no followed by a brief reason.
  
```

And *GetAction* uses the following prompt to determine the result of the action to show the player.

```

This is the story so far: [Story Context List].
[Prompt Style] + "describe what happens if I try to do
the following: [Player Action]"
  
```

4 USER STUDY

We conducted a pilot study with 16 participants to test our game, aged 21 to 47. The study was hosted online, with logging to a sqlite3 database. When participants visit the experiment URL, they are shown a title screen, followed by IRB consent information and instructions. Participants are asked to play a point-and-click interactive fiction game based on Star Trek Voyager. They are told that the game is a draft and that we need help to improve it. After a brief introductory survey that asks for age and experience with Star Trek and games, our participants play two scenes: a guided tutorial scene with open-ended dialog with two non-player characters followed by a scene in which the player is trapped in their crew quarters and need to find a way out by typing actions at a prompt. The experiment concludes with survey questions asking about the game play and opinions about generative AI.

Of our 16 participants, only 6 were familiar with Star Trek. Favorite games included "Madden Football", first-person shooters, role-playing games, and strategy. Ten out of our 16 participants completed the two puzzles and filled in the after-game surveys. Opinions about the use of AI for games were mixed. Eight out of ten participants were excited about the potential, giving feedback such as "creates a bunch of possibilities for future developments", "unlike other forms of art, I personally think ai can only enhance immersive experiences", and "it was cool - i didn't expect banter back". Two people mentioned not liking AI art, saying "Frankly,

I prefer art and entertainment made for people, BY people", and "I don't like the use of AI in story based games because I don't think it really plans ahead". Regarding feedback on the game play, four participants mentioned enjoying the conversations with our main character but ratings for understanding what to do were low (2.4/5) and three participants specifically mentioned "unclear expectations" and "confusing goals" as critiques of our demo. When asked if they would play longer if there were more puzzles on a Likert scale from 1 to 5, the average score was 3.5. Our players completed the first scene, but were confused and frustrated by the second scene, which required escaping a room using open-ended actions.

5 UTTERANCE ANALYSIS

Based on a pilot study of 16 participants and our own testing, we found that the generated responses have high coherency, moderate interestingness, but can show signs of humor and novelty. In particular, we analyzed 17 transcripts containing 160 LLM generated utterances both manually and using two automated approaches based on Chat-GPT 4.0. Coherency is a measure of how sensible generated responses are in the context of each scene. We measure coherency and interest using the LLM-based technique described by Fu *et al.* [3] which generates a score for each transcript on a scale of 1 to 5. The average coherency score across all transcripts was 4.9. The average interestingness score is 4.01. Below is an excerpt from a typical transcript. Generated utterances are shown in bold.

KELLY : Hey Rookie, are you looking forward to getting new soil samples?
 PLAYER says to KELLY: I mean, I guess?
 I haven't been trained for this.
KELLY [LLM] says: Don't worry, you'll have plenty of time to learn and get trained. Just focus on the task at hand.
 The player says to KUMAGAWA: What do you need from me?
KUMAGAWA [LLM] says: Just focus on collecting accurate and detailed soil samples, that's all I need from you.
 The player says to KUMAGAWA: Let's get those soil samples.
KUMAGAWA [LLM] says: Great, let's get to work and see what secrets the Martian soil holds.

We found several examples of humor in our transcripts. For example, when the player tried to "drink water" on the orbital Mars station, the response was "As you reach for your water bottle, John Kelly gives you a sly look and says: Careful, that's recycled astronaut pee". One player asked John Kelly who his favorite team member was. He replied "I have a great team and I couldn't choose just one, but I do have a soft spot for our rookie microbiologist".

We also check our transcripts for mistakes, both manually and with the aid of an automated model implemented with ChatGPT-4.0 that checks for different errors according to the taxonomy defined in Higashinaka *et al* [5]. This analysis revealed only 3 errors. One of wrong information, where Kumagawa says "Hello there, it's nice to have some company up here on the Ares IV space station." when she is located on the Mars surface. One of repetition, where John Kelly repeats the following sentence twice verbatim in the same conversation: "Nitroglycerine is a highly explosive substance that should only be handled by trained professionals in controlled environments". One of contradiction, where the player says they

are not a microbiologist (but they are one for the story) and John Kelly does not contradict this.

We also check for one addition error, *breaking the fourth wall*, where GPT-3.5 erroneously calls attention to the fact that we are in a game. We found one instance of this, where Kumagawa says "Is there something you wanted to discuss, Dr. [Player's Name]?" in response to no input from the player.

6 DISCUSSION AND NEXT STEPS

This work introduces a proof-of-concept narrative game consisting of both scripted and open-ended elements. Based on our user study, our early play-testers generally enjoyed the open-ended dialog with characters but expressed frustration in regards to our open-ended actions. Future work will experiment with better constraints on the action space, as well as improvements to the game interface. We are also interested in comparing our GPT-3.5 with open-source LLMs, over which game developers will have potentially greater control. These efforts have the over-arching goal of understanding what players want from generated content as well as how to design effectively with it. Even with our small demo, designing with GPT-3.5 is challenging: in addition to thinking about how the player will interact with the game, the designer must also try to anticipate how GPT-3.5 will interact with the player. An ongoing research question is how we can best leverage LLMs to anticipate and support the goals of the designer.

REFERENCES

- [1] Isaac Dart and Mark J Nelson. 2012. Smart terrain causality chains for adventure-game puzzle generation. In *2012 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 328–334.
- [2] Clara Fernández-Vara and Alec Thomson. 2012. Procedural generation of narrative puzzles in adventure games: The puzzle-dice system. In *Proceedings of the The third workshop on Procedural Content Generation in Games*. 1–6.
- [3] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. GPTScore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166* (2023).
- [4] Roberto Gallotta, Graham Todd, Marvin Zammit, Sam Earle, Antonios Liapis, Julian Togelius, and Georgios N. Yannakakis. 2024. Large Language Models and Games: A Survey and Roadmap. <https://doi.org/10.48550/arXiv.2402.18659> [cs]
- [5] Ryuichiro Higashinaka, Masahiro Araki, Hiroshi Tsukahara, and Masahiro Mizukami. 2021. Integrated taxonomy of errors in chat-oriented dialogue systems. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 89–98.
- [6] Vikram Kumaran, Jonathan Rowe, Bradford Mott, and James Lester. 2023. SceneCraft: Automating Interactive Narrative Scene Generation in Digital Games with Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 19, 1 (Oct. 2023), 86–96. <https://doi.org/10.1609/aiide.v19i1.27504>
- [7] Zhiyu Lin and Mark Riedl. 2023. An Ontology of Co-Creative AI Systems. <https://doi.org/10.48550/arXiv.2310.07472> arXiv:2310.07472 [cs]
- [8] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.
- [9] Joof Sung Park, Joseph C O'Brien, Carrie J Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442* (2023).
- [10] Reid Swanson and Andrew S. Gordon. 2012. Say Anything: Using Textual Case-Based Reasoning to Enable Open-Domain Interactive Storytelling. *ACM Transactions on Interactive Intelligent Systems* 2, 3 (Sept. 2012), 16:1–16:35. <https://doi.org/10.1145/2362394.2362398>
- [11] Mike Wollaeger, Jessica Scott, Bryan Fuller, and Michael Taylor. 1999. Star Trek: Voyager Episode 608: One Small Step. <http://www.chakoteya.net/Voyager/607.htm> Accessed July 2023.