Background/Goal

In 2012, New York’s Museum of Modern Art began archiving and showcasing famous games such as Pac-Man, Tetris, and Myst, formally arguing that these games were critical cultural artifacts worthy of preservation. As of February 2019, PCSX2 and Dolphin have seen over 10,000 and 20,000 commits to their public GitHub repositories, and have made it possible for users to upscale, upgrade, and play some of the first 3D games ever created. In 2018, Nintendo dominated Christmas shopping lists by re-releasing NES Classic, an emulator for the 1985 Nintendo Entertainment System which allowed users to easily access famous games like The Legend of Zelda, A Link to The Past. In this same year, the video games industry generated nearly 135 Billion dollars in revenue.

Given gaming’s amazing growth as a global entertainment medium, it’s saddening to see how quickly famous games are forgotten. Unlike literature, music, or film, the basic interface through which an individual can experience games is fundamentally an ever-changing thing, and it’s very difficult to entice users to engage with games of a bygone era. 40 years ago, modern story-driven games existed as Interactive Fiction: text-based adventures where players would spelunk caves and explore galaxies through simply talking with a terminal. Today, similar adventures are represented in 3D graphics rendered on a computer screen. In 10 years, these systems will likely be replaced with VR interfaces that allow users to simply walk into the worlds that their character is exploring. The games industry evolves by overriding its past, and I believe that in order to preserve classic games, these games must be accessible in a more modern form.

I hope to do this by building a conversational agent enabling users to play z-machine interactive fiction games through spoken word. Specifically, I hope to build a conversational assistant that can play Zork, the original z-machine game. Should I be successful in this endeavor, I hope to generalize this system as much as possible to run other z-machine titles. The next game on my list to incorporate would be the famously difficult Hitchhiker’s Guide to the Galaxy. Although chatbots and voice interfaces may one day grow just as stale as the terminal, I hope this

---

1 Emulators for Sony’s PlayStation 2 and Nintendo’s GameCube, respectively
2 The z-machine is a virtual machine developed by Infocom to run interactive fiction games. It is one of the most popular interactive fiction engines, and is named after Zork, the game which I hope to build a conversational system for before exploring opportunities to expand and generalize it.
research will provide a useful foundation for individuals interested in hacking, expanding, and reusing these games in the future.

Without any optimizations, the basic interface for running a z-machine game on a remote server is simple. These games were originally written to be hosted on servers and to accept and respond to text commands issued by remote hosts. Once I have the core voice→text→z-machine system built, I hope to build a few major optimizations upon the system (tracking and listing available actions, parsing natural speech) to make the games feel more accessible and modern.

**Technology Design:**

The core interaction loop for this system will be nearly identical to the Interactive Fiction games being emulated. A user will issue commands, and the proposed system will parse these commands, converting natural language speech to text, and text to a z-machine command. The system will then attempt execute these commands and will return a description of the resultant game state. Often, even well-formed commands will fail. Fortunately, these games have many hard-coded responses to failing inputs, so I do not intend to build any significant structures around command failure.

Hosting and running this system will involve three main components: 1) The Conversational Agent, 2) The Language Processor, and 3) the Game Process. **The Conversational Agent** can be easily implemented in a few ways. I can either build the agent as a “Skill” for a digital assistant such as Amazon’s Alexa, Google’s Google Assistant, or Microsoft’s Cortana, and employ one of these systems’ pre-packaged speech processing systems, or I could write an independent application that leveraged a cloud service based on these same systems, such as Amazon’s Polly Speech to Text & Text to Speech engine. I currently intend to build this as a skill for Amazon’s Alexa or Microsoft’s Cortana digital assistant. My decisioning between these two will be decided by my ability to access hardware to test the application.

**The Language Processor** will be hosted as an HTTP API on Microsoft’s Azure or Amazon’s AWS cloud servers. Ideally, I hope to decouple the language processor from the game
engine as much as possible, building my own system or leveraging a public library to extract intents from actions. I then hope to pass these basic intents to the Game Process, which will leverage a debug instance of the z-machine game to construct and issue game commands. This middle layer should be a simple process, but once functional will require significant testing to ensure that necessary commands aren’t lost in the translation process.

The Game Process will be where most of my independent code lies. I will write a master process that will receive input commands from the language processor. If these are understandable z-machine commands, the Game Process will pass them to a local instance of the z-machine game for execution, return the output, and work as a simple wrapper for the game. To add new utility to these games, the Game Process will also leverage Matthew Russotto’s Ztools to explore the game’s data and extract information such as the list of possible actions in a room (attained by decompiling the action dictionary, saving state and brute-force checking all possible actions), and to better understand the game’s text parser. For the original build of this system, I will be using a ZTools infodump, potentially alongside a txd decompilation of a specific game (Zork) to build the necessary semantic systems. I hope to explore patterns in these systems across a variety of games, with the core intent of building a maximally generalized system for constructing game commands from intents.

Should this all fit together, it should be possible to play most z-machine games (potentially with a small amount of tuning), using this conversational agent. This system, which can process a spoken command, execute it in a z-machine game, and then return to the user the z-machine response, would be my core deliverable.

Additional Features:
Should work finish early, I would love to build a rich chatbot interface for playing these games. The inputs and outputs from the game engine will be strings, but if through debug systems I can pull the set of possible actions in a room, or root out the set of additional rooms connected to a current position, I could create an action-suggestion card system for the game.

Building such a system based solely on debug output from a text-based adventure might be difficult, but it would be an interesting challenge should I have time and resources.