Using Microphones to Control a RPG Game

Henry Wilt

Ursinus College
hewilt@ursinus.edu

February 10, 2022

Abstract

The goal for the assignment was to incorporate a Speech Recognition Program created by Dr. Mongan [1]. We had the option to write either a calendar app that finds come times between people or to use Voice and text to speech to create a role-playing game that the player explores a maze of connected rooms. In my program, I utilized voice controls from the computers microphone to control how the player would react in the world. There is no need for a keyboard, mouse, or visual clues to play the RPG game.

I. About the Game

The game is a simple escape the maze sort of RPG; the game is set in a cave. The starting location is in the back of the cave and the goal for the player to do is to escape the cave. There are treasures in the cave that you can collect in some of the cave rooms. There is also a monster that you can find in the cave and beat to gain a way out of the cave.

i. Code Structure

The code for the assignment can be found at my GitHub [2]. There are two classes that are necessary to run the game, "main.py" and "player.py". We will start out in the player class, where it is pretty simple when you initialize a new player object it will hold member data such as name, health, and the location of the player. Next, the class is full of methods that allow the programmer to access all the player's member data; such to set the health if damage is taken or change the name of player. The main class contains the bulk of the code that run the actual game. It has the voice class which allows the microphone to pick up the voice of the player and speak to the player with text to speech. It also contains helper methods to run the game such as a direction method, that asks the user which direction it wants to go, and a pickup method, that asks the user if they want to pickup the item in the room they are in.

II. Considerations

In this assignment, Dr. Mongan wants us to consider how will the user know they should say a response? How will I handle speech recognition errors? If the speech program picks its own prompts up during recognition, how would I address it? How should the program be configured for appropriate delays for your user's responses?

In my program, every time the voice class's listen method, it will "speak" to the user and say "Please Speak Now" to show the user that after the computer says that they will then respond. The listen method then picks up on the voice and returns the dictionary to the call. For handling speech recognition errors, I keep calling the listen method which will tell the user that the computer could not understand them. When it is called again, the user will have another chance to respond to the computer. The speech program can not pick up it's own prompts when waiting for the user to respond, I addressed this by waiting a second every time the listen method is called and

also when the text to speech is going off, the program will wait for it to finish til it says the whole message then it will continue. I found the appropriate delay to be around 0.5 to 1 for the people I tested the program on. If you wait too long then the user will find it annoying that the computer is not listening to them and if you don't wait enough the user will be mad that they could not get their response in before it told them they didn't hear them or didn't get the response. So I settled on 0.5 as it is not too quick that the computer is missing the user's response but it is not too slow the user is having to repeat themselves so the program can hear them.

III. Conclusions

The assignment was fun and interesting. I learned a lot about speech recognition and what problems it can fix. Even though I made a game for my assignment, I can see the possibilities of using speech recognition for a whole lot more like to create calendar events, taking notes in the classroom for people who don't write as fast, and could be used to benefit people who are deaf as a speech to text translator.

References

[1] Bill Mongan. *CS474: Human Computer Interaction: Voice Prompts Assignment Page*. URL: https://www.billmongan.com/Ursinus-CS474-Spring2022/Assignments/Programming/VoicePrompt. (accessed: 02.10.2022).

[2] Henry Wilt. *Voice Prompts Assignment Code*. URL: https://github.com/hwilt/CS474/tree/main/Voice_Prompts. (accessed: 02.10.2022).