Voice Controlled Personal Assistant Using Raspberry Pi

Ass. Prof. Emad S. Othman, Senior Member IEEE - Region 8, High Institute for
Computers and Information Systems, AL-Shorouk Academy, Cairo – Egypt,
PH- 0020-010-25830256. E-mail: *emad91@hotmail.com*

Abstract–**The purpose of this paper is to illustrate the implementation of a Voice Command System as an Intelligent Personal Assistant (IPA) that can perform numerous tasks or services for an individual. These tasks or services are based on user input, location awareness, and the ability to access information from a variety of online sources (such as weather or traffic conditions, news, stock prices, user schedules, retail prices, telling time, local traffic, travel assistant, events, notification from social applications plus one can ask questions to the system, invoke its machine learning otherwise get it from Wikipedia… etc…..).**

Using Raspberry Pi as a main hardware to implement this model which works on the primary input of a user's voice. Using voice as an input to convert into text using a speech to text engine. The text hence produced was used for query processing and fetching relevant information. When the information was fetched, it then be converted to speech using text to speech conversion and the relevant output to the user was given. Additionally, some extra modules were also implemented which worked on the concept of keyword matching.

It can help the visually impaired to connect with the world by giving them access to Wikipedia, Calculator, Email and Music all through their voice. This model can also keep people secure as it can be used as a surveillance system which captures the voice of the person standing at the door and similarity checking. Also it can be a source of entertainment and information for blind/visually impaired. This model will interact with other systems by means of IOT, thus provides a fully automated system. Many experiments and results were accomplished and documented.

Keywords-- Personal Assistant, Text to speech, Speech to text, Raspberry Pi, Voice Command System, Query Processing Machine Learning.

I. INTRODUCTION

A Voice Command System essentially means a system that processes voice as an input, decodes or understands the meaning of that input processes it and generates an appropriate voice output. Any voice command system need three basic components which are speech to text converter, query processor and a text to speech converter. Voice has been a very integral part of communication nowadays. Since, it is faster to process sound and voices than to process written text,

hence voice command systems are omnipresent in computer devices.

There have been some very good innovations in the field of speech recognition. Some of the latest innovations have been due to the improvements and high usage of big data and deep learning in this field [1]. These innovations have attributed to the technology industry using deep learning methods in making and using some of the speech recognition systems, Google was able to reduce word error rate by 6% to 10% relative, for the system that had the word error rate of 17% to 52% [2].

Text to speech conversion is the process of converting a machine recognized text into any language which could be identified by a speaker when the text is read out loud. It is two step processes which is divided into front end and back end [3].

First part is responsible for converting numbers and abbreviations to a written word format. This is also referred to as normalization of text. Second part involves the signal to be processed into an understandable one [4].

Speech Recognition is the ability of machine for instance a computer to understand words and sentences spoken in any language [5]. These words or sentences are then converted to a format that could be understood by the machine. Speech recognition is basically implemented using vocabulary systems [6]. A speech recognition system may be a Small Vocabulary-many user system or a Large Vocabulary- small user system [7].

II. SYSTEM ARCHITECTURE

Existing Systems

The existing systems suffer from the drawback that only predefined voices are possible and it can store only limited voices. Hence, the user can't get the full information coherently.

Proposed System

The proposed system is such that it can overcome the drawback of the existing system. The project design involve text to speech. Here whatever the system receives as input

after the command the output will get in the form of voice means speech.

Hardware Implementation

Microphone is used to take the audio input of the sound. This audio input when further passed through the system would be searched for keywords. These keywords are essential for the functioning of the voice command system as our modules work on the essence of searching for keywords and giving output by matching keywords as shown in Figure 1.

Keyboard acts as an input interface mainly for the developers, providing access to make edits to the program code.

Mouse also acts an interface between the system and the developer and does not have a direct interaction with the end user.

Raspberry Pi is the heart of the voice command system as it is involved in every step of processing data to connecting components together [8]. The Raspbian OS is mounted onto the SD card which is then loaded in the card slot to provide a functioning operating system.

The Raspberry Pi needs a constant 5V, 2.1 mA power supply. This can either be provided through an AC supply using a micro USB charger or through a power bank.

Ethernet is being used to provide internet connection to the voice command system. Since the system relies on online text to speech conversion, online query processing and online speech to text conversion hence we need a constant connection to achieve all this.

Monitor provides the developer an additional way to look at the code and make any edits if any. It is not required for any sort of communication with the end user.

Speakers, once the query put forward by the user has been processed, the text output of that query is converted to speech using the online text to speech converter. Now this speech which is the audio output is sent to the user using the speakers which are running on audio out.
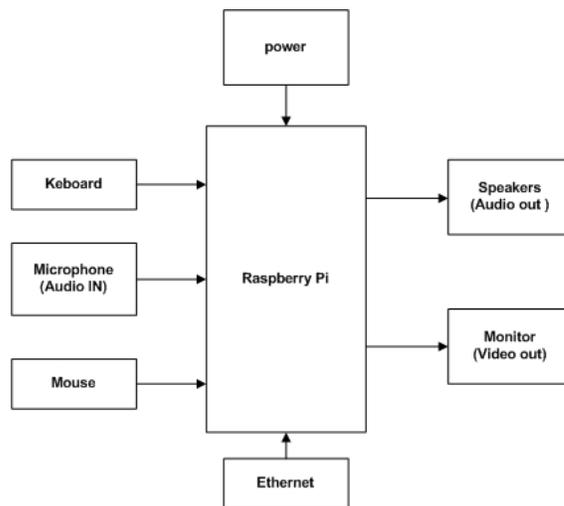
Figure 1. Hardware setup of voice command system

Flow of Events in Voice Command System

First, when the user starts the system, he uses a microphone to send in the input. Basically, what it does is that it takes sound input from the user and it is fed to the computer to process it further. Then, that sound input if fed to the speech to text converter, which converts audio input to text output which is recognizable by the computer and can also be processed by it.

Then that text is parsed and searched for keywords. Our voice command system is built around the system of keywords where it searches the text for key words to match. And once key words are matched then it gives the relevant output.

This output is in the form of text. This is then converted to speech output using a text to speech converter which involves using an optical character recognition system. OCR categorizes and identifies the text and then the text to speech engine converts it to the audio output. This output is transmitted via the speakers which are connected to the audio jack of the raspberry pi as shown in Figure 2.
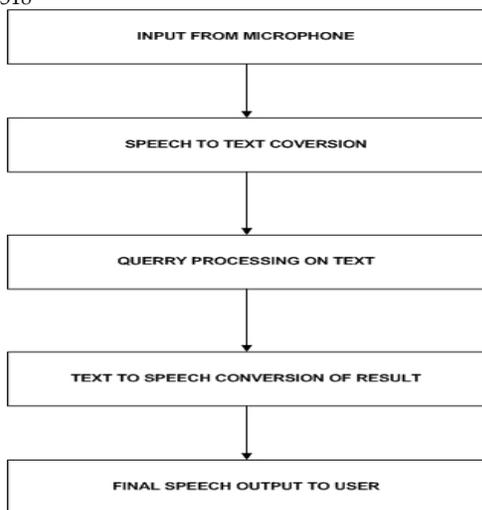
Figure 2. Flow of events in voice command system

III. Modules Implemented

Speech To Text Engine

(AVS) is a Speech-To-Text (STT) engine which is used to convert the commands given by the user in audio input to text form, so that these commands can be interpreted by the modules properly. To use (AVS) engine, an application has to be created in the Amazon developers console and the generated API key has to be used to access the speech engine. It requires continuous internet connection as data is sent over the Amazon servers.

Text To Speech Engine

(AVS) is a Text-To-Speech (TTS) engine is used to create a spoken sound version of the text in a computer document, such as a help file or a Web page. TTS can enable the reading of computer display information for the visually challenged person, or may simply be used to augment the reading of a text message. To use (AVS) engine, an application has to be created in the Amazon developers console and the generated API key has to be used to access the speech engine. It requires continuous internet connection as data is sent over the Amazon servers.

Query Processor

The Voice Command System has a module for query processing which works in general like many query processors do. That means, taking the input from the users, searching for relevant outputs and then presenting the user with the appropriate output. In this system we are using the site wolfram alpha as the source for implementing query processing in the system. The queries that can be passed to this module include retrieving information about famous

personalities, simple mathematical calculations, description of any general object etc.

Wikipedia

This module works on the keyword of "wiki". The system asks for what you would like to learn about. Then the request is made to the Wikipedia API for the required query. It generates the summary of the information regarding the query and the data is output through the microphone to the listener in audio form. In case of failure, the error message is generated saying "unable to reach dictionary of wiki".

News

News module can be executed by using the keyword "news". The headlines of top articles are retrieved from the internet using Google news. The system tells the user all these headlines and asks the user if any of these articles should be sent to the user's email address. If the user specifies the number of the article to be sent, the article is sent to the specified email address. Otherwise, no further action is taken. If any failure occurs in retrieving headlines or sending articles, a corresponding error message is generated.

Weather

This module tells the user about the weather conditions of the location whose station identifier is specified in the profile of the user. This module can be executed by using the keyword "weather". The weather information is taken from the weather underground service which includes the details of temperature, wind speed and direction etc. It generates an error message, if the information cannot be retrieved for the specified location.

Movies

The voice command system searches for relevant movie by the keyword "movie". It is implemented using a function which asks the user, the movie they want to know more about. Then a function is called to ask for the movie the user needs to know about. It searches for the top five results regarding the movie name. It confirms the movie among the listed names. On confirmation of the movie, it gives the detailed information including the rating, producer, director, cast, runtime and genres. In case of the failure, it generates the error of "Unable to find information on the requested movie".

Define

Define module is used to fetch the definitions of word specified by the user. The execution of this module is started by the keyword "define". Then, the user is asked for the word whose definition has to be provided. The module checks for the definitions of the specified word using the Yandex Dictionary API. All the available definitions are provided as a

response from the API and the same is given as output to the user in audio form. If the connection to API cannot be established properly, then the error message "Unable to reach dictionary API" is generated for the user.

Find My IPhone

Find my iPhone module lets the users find their iPhone by their voice. This is done using the user's iCloud id and password to make the connection to the server. After the authentication of the password it checks the status of the iPhone. After checking, it connects to iPhone using the iCloud. After the connection, the command for ringing is sent to the iPhone to start a sound and notification on the phone for detection. If a failure occurs, the system generates an error that the iPhone is not found. An error is also generated when there are multiple iPhones associated with a single apple id.

Joke

Joke module can be used for entertainment purposes by the user. This module works on the keywords "joke" or "knock knock". The jokes used in this module are predefined in a text file from which the jokes are read in a random order. A start and end line is present in every joke to differentiate it from others present in the file. All the lines of a joke are spoken by the system in the specified order only.

Unclear

This module is used to generate an error message if the keyword specified by the user does not match the keywords present in any module of the system. This module has an empty array of keywords and is allocated the least priority, so that it is executed only after the keyword has been matched with all the other modules of the system. The messages generated by this module include statements like "I beg your pardon", "Say that again", "I'm sorry, Could you repeat that" and "My apologies, Could you try saying that again" which are selected in a random order each time this module is executed.

Other Command Specific Modules

The Voice Command System also has some command specific modules like fetching hacker news, email and current time. Each of these modules is related to the system using keywords like "hacker news", "email" and "time" respectively. Whenever any of this keyword is said to the system, it fetches that module and launches the contents of that module thereby providing the appropriate response to the user.

IV. RESULT

The Voice Command System works on the idea and the logic it was designed with. Our personal assistant uses the button to take a command. Each of the commands given to it is matched with the names of the modules written in the program code. If the name of the command matches with any set of keywords, then those set of actions are performed by the Voice Command System. The modules of Find my iPhone, Wikipedia and Movies are based upon API calling. We have used open source text to speech and speech to text converters which provide us the features of customizability. If the system is unable to match any of the said commands with the provided keywords for each command, then the system apologizes for not able to perform the said task. All in all, the system works on the expected lines with all the features that were initially proposed. Additionally, the system also provides enough promise for the future as it is highly customizable and new modules can be added any time without disturbing the working of current modules.

V. CONCLUSIONS AND FUTURE WORK

In this paper, introduced the idea and rationale behind the Voice Command System, the flaws in the current system and the way of resolving those flaws and laid out the system architecture of the presented Voice Command System. Many modules are of open source systems and have customized those modules according to the presented system. This helps get the best performance from the system in terms of space time complexity.

The Voice Command System has an enormous scope in the future. Like Siri, Google Now and Cortana become popular in the mobile industry. This makes the transition smooth to a complete voice command system. Additionally, this also paves way for a Connected Home using Internet of Things, voice command system and computer vision.

Acknowledgment

References

[1] Dahl, George E., et al. "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition." Audio, Speech, and Language Processing, IEEE Transactions on 20.1 (2012): 30-42.

[2] Chelba, Ciprian, et al. "Large scale language modeling in automatic speech recognition." arXiv preprint arXiv:1210.8440 (2012).

[3] Schultz, Tanja, Ngoc Thang Vu, and Tim Schlippe. "GlobalPhone: A multilingual text & speech database in 20 languages." Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.

[4] Tokuda, Keiichi, et al. "Speech synthesis based on hidden Markov models."Proceedings of the IEEE 101.5 (2013): 1234-1252.

[5] Singh, Bhupinder, Neha Kapur, and Puneet Kaur. "Speech recognition with hidden Markov model: a review." International Journal of Advanced Research in Computer Science and Software Engineering 2.3 (2012).

[6] Lamere, Paul, et al. "The CMU SPHINX-4 speech recognition system." IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong. Vol. 1. 2003.

[7] Black, Alan W., and Kevin A. Lenzo. "Flite: a small fast run-time synthesis engine." 4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis. 2001.

[8] Lee, Chin-Hui, Frank K. Soong, and Kuldip Paliwal, eds. Automatic speech and speaker recognition: advanced topics. Vol. 355. Springer Science & Business Media, 2012.

[9] Elaine Rich, Kelvin Knight, Shiva Shankar B-Nair, "Artificial Intelligence", MC Graw Hill Publication, "2013".

[10] Steven Hickson, "Voice Control and Face Recognition on the Raspberry Pi"[online], Available at http://stevenhickson.blogspot.com.

[11] Raspbian[online], Available at http://www.raspbian.org/.

[12] Wayne Wobcke, Anh Nguyen, Van Ho and Alfred Kqzywicki, School of computer science and engineering, University of new south walls, Sydney nsw 2052, Australia, "the smart personal assistant: an overview". (jounal[online])

[13] "Intelligent Personal Assistant", "siri", "corrana", [online] by Wikipedia.

[14] "Pineapple and Raspberry Pi", circuit designs and comparison in google".

IJSER