Translating System with Android and Raspberry PI

Jae Moon Lee, In-Hwan Jung, Kitae Hwang

Abstract Background/Objectives: ***Recently, translation systems have been focused due to growth of web services and smart phones. This paper aims to develop noble translation system with smart phone to IoT device.***

Methods/Statistical analysis: ***The proposed system uses a smart phone and an IoT device for translation. The IoT device works simply to transmit recorded audio to smart phones and to receive and display the translated context. The smart phone requests to translation through web services. The smart phone also transmits result to an IoT device or displays it. An advantage of such methods is that it requires minimum function from the IoT.***

Findings: ***An Android smart phone was used to develop this system. Raspberry pi was used as the relevant IoT device. Both devices recorded audio with the same method. Recording was conducting through a PCM method. Sampling Rates were selected as variable, so that the size of the recording file could be selected as needed. Raspberry pi delivers the recorded audio to the Android smart phone via bluetooth. And it receives the translated text, and displays it. The smart phone converts the recorded/received audio file to text, and translates it into a desired language. Google Speech Service was used to convert the audio file to text. The Naver Clova was used to translate the text. Both web services have applied Artificial Intelligence technology, and thus yielded results of satisfactory quality.***

Improvements/Applications: ***The greatest advantage of the proposed system is the minimized function of the IoT device. Thus, it is possible to develop the translating system with minimum cost.***

Keywords: Smart Phone, IoT, Web Services, Speech to Text, Translation.

I. INTRODUCTION

Through recent technological advancements, runtime translation has been achieved and the quality has also improved greatly. With such advancements, many systems that offer runtime translation have been developed [1,2,3]. Also, through the improvement of artificial intelligence technology, the quality of translation continues to improve. Audio speech to text and translation of texts are happening runtime through utilizing web service techniques. Especially, smart phone applications that can manage simple conversation translations are already being commercialized [3,4]. This paper aims to develop a runtime translation device that can handle such simple level conversations.

Revised Manuscript Received on May 22, 2019.

Jae Moon Lee, In-Hwan Jung, Kitae Hwang, School of Computer Engineering, Hansung University, 116 Samseongyoro-16Gil, Seongbuk-Gu, Seoul, 02876 Korea

E-mail: jmlee@hansung.ac.kr

Translation, in essence, is required by people engaging in a conversation of two or more people. Numerous translation applications are being developed, but a few fundamental obstacles exist that limit their use in a real-life circumstances. The first is that, in order for a smart phone to translate, one must speak, wait for the translation, show the other person the translation, and then wait for the other person to repeat this process every time they speak. A solution to this installs the same translating application on two smart phones and then each person participating in the conversation translates his/her words respectively and exchanges his/her translated words. However, the solution is not realistically viable in that each person must download the identical application, and set up network preferences. Another solution to the problem would be for one to carry two separate smart phone devices. However, whether or not carrying two smart phone devices for the sake of a simple conversation translation is worth it may be controversial.

This paper aims to develop a translation device that will connect to smart phones, in order to overcome such problems that challenge the conventional translation system. The desired system will utilize an IoT (Internet of things) device for translation. This device will not require further internet resources or translation resources, since it will be connected to a smart phone. The device shares a smart phone's internet/translation resource to achieve functions.

II. RELATED WORKS

Smart phone technology is pioneering all computing technologies. This is because the smart phone has long surpassed the original use as a simple phone, and now computing ability has overpowered the conventional use [5]. Voice recognition, artificial intelligence, and virtual reality are the epitome of such computing ability [2,6,7,8]. Apple has recently revealed a kit to aid the development of applications that can actualize virtual reality technology. Voice recognition technology have been developing into web service technology, utilizing smart phones [3,4]. Technologies for such web services have been developed under the lead of massive portal sites such as Naver [3] and Google [4]. The numerous smart phone applications that utilize such services through network are also under spotlight. The proposed system also utilizes such technologies.

IoT, short for Internet of Things, is a new technology that connects various "things" to the internet through sensors and communication

technology [6.9]. Thus, it refers to the technology of connecting objects through wireless communication. It allows objects that are connected to the Internet to provide to the users the information it has analyzed and learnt through transmitting and receiving data. It also allows the users to remotely control the information. These "objects" or "things" refer to various embedded systems such as electronics, mobile devices, and wearable devices. One typical example of an IoT device is the smart phone. Various technologies such as low-power networking, optimization and management of sensor data, low-power embedded OS technology, new power supply and storage, low-cost/low-power are currently being developed to bring IoT technology to life. Especially, the low-power provision and storage allows a device to last as long as possible with limited resources, and is the core technology of IoT. The embedded system is a representative product of the IoT technology. The embedded system is an electronic system that embeds or includes all computer system to control necessary machines or systems [10,11]. One thing to note about the embedded system is that it offers both the hardware and the operating system. Also, the offered operating system supports runtime processing. Currently, the most commonly used embedded system is the Arduino and Raspberry pi [6,9,10,11]. Arduino refers to a board, related development tools and environment completed through a single board micro-controller based on an open source. Arduino was designed in 2005 by students who were unfamiliar with the hardware to easily control their design works. Currently, the AVR board is its best seller. There are also products that use the AMR Cortex-M0. The Raspberry pi is a single board computer with the size of a credit card developed by the Raspberry pi foundation of England, to promote computer science education in schools and the Third World countries. Because the Raspberry pi surpasses by far Arduino in terms of function and performance, this paper has adopted the Raspberry pi as the IoT device.

Speech Recognition refers to the process of a computer converting verbal language into text data. It is also referred to as STT (Speech-to-Text) [1.2.3]. STT has been spotlighted as an alternative means of text input, against the conventional method of using a keyboard. STT can be applied in areas such as robots and telematics where device control and info search is needed. The HMM (Hidden Markov Model) [12] is a key algorithm for STT. It statistically models numerous audios of numerous speakers to form a sound model. Clova Speech Recognition API, developed and serviced by Naver, inputs the voice of user in a streaming form and converts the results into text [3,7,8]. Naver has developed independently streaming protocol to transmit the audio of user. Thus, the API is provided in Android and iOS SDK format, instead of the HTTP based REST API format. Google provides the Google Cloud Speech-to-Text that applies a strong neural network model to the developers who convert audio to text [4,13]. This service promises significantly improved voice recognition performance. The API promises a reduction in word errors around 54 percent across all of Google's tests, but in some areas the results are actually far better than that. This API recognizes over 120 languages and dialects, in correspondence to the global user. Using Google's Machine Learning technology, runtime streaming or pre-recorded audio can be processed. Naver solely supports streaming, whereas Google supports the upload of audio files as well as streaming. Thus, this paper has adopted the more suitable Google Speech Service.

Naver and Google are also the two major websites that provide translation services. Papago is the free translation service that Naver provides [3]. It is an Artificial Neural Network based translation service that Naver has developed, and can provide a total of 30 combinations of translation within six different languages: Korean, English, Japanese, Chinese, French, and Spanish. Currently, Papago is provided for free on Android and iOS devices, as well as PC. Google Translator is a multi-lingual translation service provided for free by Google [14]. It is a web application designed to allow translators to edit the translations that Google Translate automatically generates. The Google Translation API provides a simple programmatic interface for translating arbitrary strings into supported languages using state of the art artificial neural network translations. Because it is responsive, the developer can integrate websites and applications with the Translation API to quickly and dynamically translate the source text of the source language into the destination language. With the Google Translator, translators can organize their work and use shared translations, glossaries and translation memories. It provides API that aids the development of application software in Android, iOS, website interface, mobile apps. It can translate sentences, paragraphs, documents, and web pages. Google Translate supports over 100 different languages at different levels, and is processing over 500,000,000 requests every day, as of May 2017. However, this paper will use Naver Papago service, for the sake of system diversity.

III. DESIGN AND IMPLEMENTATION

3.1. Design

The developing translation system will be composed of an Android smart phone and Raspberry pi. The Android phone operates on a server, and the Raspberry pi operates on a client. The smart phone and Raspberry pi will communicate in a server-client perspective. The translation system works in the following procedure: first, the Raspberry pi will record the voice of the user when he/she speaks, and will deliver it to the server- the smart phone. The smart phone will then translate the transmitted audio file through the Internet and display the translated content on the smart phone. Secondly, when the user of the smart phone is speaking, the smart phone will record the voice of the user and translate it through the Internet. The translated content will be sent to Raspberry pi, where it is displayed.

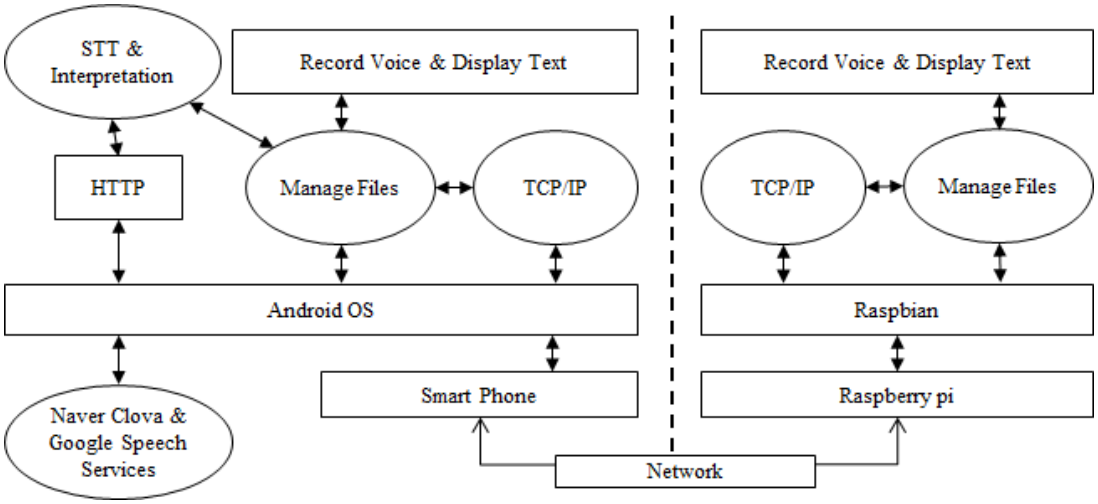

Figure 1. Architecture of System and Core Modules

Smart phone	Raspberry pi
1) Voice of user is recorded and saved as a file 2) The recorded audio is converted into a text file through Google's Speech service 3) The converted text is translated through Never Clova 4) The translated Text is transmitted to the client through bluetooth	1) Receives text from server 2) Displays the received text

Figure2. Scenario 1: Translation and Transmission of user's voice in server side

Smart phone	Raspberry pi
1) The audio file is received from the client 2) The received file is converted to Text through Google's Speech service 3) The converted text is translated through Never Clova 4) The translated text is displayed	1) Voice of user is recorded and saved as a file 2) The recorded audio file is transmitted to the server

Figure 3. Scenario2: Translation and Transmission of user's voice in client side

To maximize efficiency, this paper has designed the system as can be seen in [Figure 1]. The system has been composed with an Android phone and Raspberry pi. Both components have transmission functions of files. Also, both components will be able to record audio and display text. The Android phone can connect to the Internet, and will translate texts through it. The greatest advantage of this system is that Raspberry pi, the client, does not need to connect to the Internet.

As can be seen in [Figure 1], bluetooth is utilized as the necessary means of network. Most smart phones have a built in bluetooth function. Raspberry pi also has a built in bluetooth function for short distance communication. This system uses TCP/IP based on bluetooth. Here, TCP/IP is a protocol used for file transmission. Because communication in this system is a simple one-on-one transaction between two devices, issues such as security is not a major factor to be put into consideration. The file manager module saves the recorded audio as a file, or sends the recording files to other devices.

The Google Speech Service, which is an STT module, is a module that converts the recorded audio files into text files. In this system, any recorded or received files will first be converted into a text file. The module performs such conversions. A translation module reads the text file and then translates it to a target language. To be more specific, it reads the content of the text file, requests translation to an external translation service, and receives the results. Recently, many have been providing translation as a form of Cloud Service. The "Record Voice & Display Text" simply records the user's conversation, saves the file, and displays the translated text.

Two translating scenarios are shown in [Figure 2, Figure 3]. The scenario 1 regards a situation where the Server user's audio is delivered to the client. The scenario 2 regards a situation where the Client user's audio is delivered to the server.

3.2. Implementation

The Android operating system was used for the server - the smart phone. The SDK of the Android operating system was implemented through Android Studio 3.0, provided by Google. The main development language for the Android Studio is Java. Most libraries needed for the development of the proposed system are provided in Java libraries, and thus it is sufficient to be developed through Java. Bluetooth was used to provide a network environment. Google Speech Service [2] was used as the necessary technology to convert audio to text, and the Naver's Clova [3] was used as the necessary translation tool. Other functions were used from the Android operating system. Raspberry pi was used as the IoT device and client [6]. The Raspbian operating system was used as the operating system for this device. The Raspbian is an operating system created by the developers of Raspberry pi. Also, it is the official Raspberry pi operating system and a Debian Linux distribution version. Java in the client was adopted as the development language. This was so reconciled the development language with that of the server, and also because it allows the system to easily be adopted by other platforms. The libraries of recording voice were used one provided in Java. Other functions were implemented by using the original libraries provided in the Raspbian.

The recording module records the user's audio. It is appropriate for recording at the low level. The recordings were recorded in PCM format. In terms of recording, there is always a trade-off between the quality of the recording and the size of the file. Thus, it is under the user's discretion to choose amongst 8000, 11025, 16000, 22050, 32000, 44100 sampling rate. The operation of an audio recording module is already well known. Most SDKs support an audio recording object. Android operating system supports the AudioRecord object. It operates by firstly creating the object and continuing to read out and save the recording data produced by the object through thread, etc. It is crucial that the threat object catches all the data produced by the object.

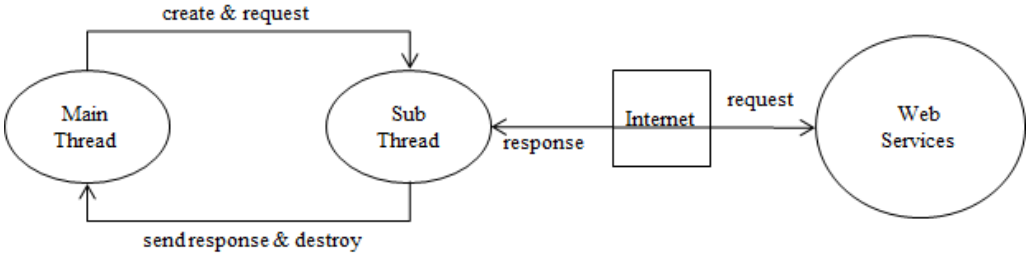

Figure 4. Implementing Method of General Access for Web Services

The STT module and Interpretation module uses web services. The STT module uses Google's Speech service, and The Interpretation module uses Naver Clova service. The most implementations of using web services are very similar. As can be seen from [Figure 4], the main thread first creates a sub thread which can perform the needed tasks. It delivers the necessary data and requests the sub thread to perform the needed tasks. The sub thread will format the data in appropriate forms and request a web service through Http. The sub thread waits until the results have been delivered, and expires after conveying the end product to the main thread. The STT module sends the audio file to Speech Service which is a part of Google's Cloud Service and receives converted texts. The Speech Service is one of Google's paid services. Thus, one must apply for service at Google and when receiving a credential JSON file upon request, it must be included in the project. The realization of this module is fairly simple. It can be achieved simply by utilizing Google's SpeechService class. The operation of the class is also realized in methods that are seen in [Figure 4]. To use this, one can simply register the sample rate, and input the PCM file. Once that is done, a callback function is summoned like most other cloud services. The converted files can be read in the callback function. The translation module is a module that delivers texts to Naver Clova, and receives the results. It is serviced in ways that are similar to that of Google. One must apply for service to Naver, and apply the API Key provided upon request to the program, and send it to Naver Clova if necessary. Naver does not provide any class for the service of Naver Clova. Thus, the developer must create a class using methods seen in [Figure 4]. The most common way to realize this class is to use threads. The thread is given a text file that requires translation. It will then request translation through Http, and wait until the results are received. When the results are delivered, it notifies the main thread. To do this, the proposed system utilizes the AsyncTask class provided by Android libraries.

3.3. Performance Measurement

The performance of the proposed system be evaluated by largely two standards. The first is the quality of the translation, and the second is the runtime capacity. The former assesses the accuracy of the translation and the accuracy of the conversion of audio to text. The later assesses the speed of network, Google Speech Service, and Naver Clova's translation. Amongst these factors, the accuracy of converting audio to text, translation, speed of Google Speech Service and Naver Clova's translation cannot be enhanced through the proposed systems. Thus, the proposed system aims to enhance performance by seeking the minimum size of the PCM file that will guarantee correct translation. These experiments were conducted in Samsung Galaxy S7 and LTE environment.

Table 1: Response times for STT versus recording times

Recording Time	5sec	10sec	20sec
File Size	433KB	864KB	1,725KB
STT Time	3,393msec	4,479msec	6,380msec

The relationships between time to convert STT and the

measured file sizes according to recording times are shown in [Table 1], where the sampling rate was fixed at 44,100Hz. As shown in Table 1, the size of the file increases linearly as the recording time increases. That is, if the recording time is doubled, the file size is doubled. But the STT time is not. This is because the time required for the preparation for conversion is constant regardless of the file size. From Table 1, it is not difficult to calculate that this initialization time is about 2300 msec. It can be seen that if 2300 msec is subtracted for each STT Time in Table 1, it increases almost linearly with the recording time. However, the initialization cost is a characteristic of Google's SpeechService, so it is not a problem that can be solved in this study. According to these results, it is efficient to convert large files that have been recorded for as long as possible.

Table 2: Response times for STT versus sampling ratio

Sampling Rate	6,000Hz	8,000Hz	16,000Hz	32,000Hz
File Size	294KB	392KB	784KB	1,567KB
STT Time	N/A	6,072msec	6,436msec	7,609msec

The relationships between time to convert STT and the measured file sizes according to sampling ratio are shown in [Table 2], where the recording time was fixed at 25 sec. According to the structure of the PCM, the file size is linearly proportional to the sampling rate. In our experiment, the Google Speech Service cannot convert when the sampling rate is 6000Hz. It is judged that the sampling rate is too low to recognize the voice. However, from 8,000 Hz and above, the conversion result was very accurate. As shown in Table 2, the file size increases linearly with increasing sampling rate, but the response time is not. Also, the response time difference is very small. From this experiment, it can be seen that the sampling rate is not a critical factor in the performance of the developed system.

IV. CONCLUSION

Smart phones have been improved rapidly. People are now taking care of numerous tasks on their smart phones that are connected to various web services. A prime example is runtime translation. Along with the development of smart phones, IoT technology has been gathering great attention. This technology, based on low-power technology, provides various services in mobile forms. Raspberry pi is a great example of the IoT system. This paper proposed a new translation system that integrates smart phone functions and IoT systems.

The proposed system was implemented as a prototype, and is currently under various performance enhancements and measurements. The key technology to the proposed system is to record an audio, deliver the recorded file to be translated, and to receive the file in text format. The greatest overhead is the size of the file. If the file is big, an accurate text is obtained but the speed is slowed down. In the opposite case, the conversion audio to text text may not be accurate.

ACKNOWLEDGMENT

This work was financially supported by Hansung University.

REFERENCES

1. Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields. Cocaine noodles: exploiting the gap between human and machine speech recognition. WOOT. 2015; 15: 10-1.
2. Schuster, Mike. Speech recognition for mobile devices at Google. Pacific Rim International Conference on Artificial Intelligence, Springer, Berlin, Heidelberg. 2010; 8-10.
3. Kang Inho. Clova: Services and Devices Powered by AI. The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM. 2018; 1359-1359.
4. J Twiefel, T Baumann, S Heinrich, S Wermter. Improving Domain-independent Cloud-Based Speech Recognition with Domain-Dependent Phonetic Post-Processing. AAAI. 2014; 1529-36.
5. Zulstra Judith, Liempt Ilse Van. Smart (phone) travelling: Understanding the use and impact of mobile technology on irregular migration journeys. International Journal of Migration and Border Studies. 2017; 3(2): 174-91.
6. Saad Ali Younis, UmerIjaz, Ahmad Randhawa, AbubakerIjaz. Speech Recognition Based Home Automation System using Raspberry Pi and Zigbee. NFC IEFR Journal of Engineering and Scientific Research. 2017; 5: 40-5.
7. Park Geonwoo; Kim Harksoo. Low-Cost Implementation of a Named Entity Recognition System for Voice-Activated Human-Appliance Interfaces in a Smart Home. Sustainability. 2018; 10(2): 1-11.
8. Adrian Kim, Soram Park, Jangyeon Park, Jung-Woo Ha, Taegyun Kwon, Juhan Nam. Automatic DJ mix generation using highlight detection. The 18th International Society for Musical Information Retrieval Conference (ISMIR). ISMIR. 2017; 1-2
9. Lakshmi, K., Rao Mr T. Chandra Sekhar. Design and implementation of text to speech conversion using Raspberry Pi. IJITR. 2016; 4(6): 4564-7.
10. Amit Mishra, DipakPatil, Nikhil Karkhanis, VaishnaviGaikar. KadambariWanil. Real time emotion detection from speech using Raspberry Pi 3. Wireless Communications, Signal Processing and Networking (WiSPNET) 2017 International Conference on IEEE. 2017: 2300-3.
11. M. Rajesh, BindhuRajan, Ajay Roy, Almaria Thomas, Ancy Thomas, BincyTharakan, et al. Text recognition and face detection aid for visually impaired person using Raspberry PI. Circuit, Power and Computing Technologies (ICCPCT), 2017 International Conference on IEEE. 2017: 1-5.
12. Anders Krogh, Björn Larsson, GunnarvonHeijne, Erik L LSonnhammer. Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. Journal of molecular biology. 2001; 305(3): 567-80.
13. ManaswiNavin Kumar. Speech to Text and Vice Versa: Deep Learning with Applications Using Python. Apress: Berkeley, CA; 2018. 127-44.
14. Chand Sunita. Empirical survey of machine translation tools. Research in Computational Intelligence and Communication Networks (ICRCICN), 2016 Second International Conference on IEEE. 2016; 181-5.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication