Smart Stick for Visually Impaired Users

Abstract

For common applications in Internet of Things (IoT), the purpose aims at improving working efficiency or facilitating user interactions in various degrees of their daily life. We implemented this concept through creating a Blind Stick for visually impaired users that helped them navigate to new locations outside their home. This was accomplished through the use of a Raspberry Pi acting as the Smart Stick and communicating with an Android smartphone. The Smart Stick will provide the user with tools needed to navigate their direction and the ability to request help from a close friend or family member.

Key Words - Smart Stick, Raspberry Pi, navigating

Introduction

During the collecting process of related materials, it was found that most IoT applications on blind stick concentrate on helping users detecting the obstacle on their way. And this function could usually be realized with an ultrasonic module integrated on various microprocessors, like Arduino or Raspberry Pi. Under such circumstance, we decided to design and implement a novel function to highlight the uniqueness and creativity of our project. Thus the idea of navigating came out.

The objective of this project is to help navigate visually impaired users to their interested destination of choice. Meanwhile, if there occurs any emergency on the user, he/she can send predefined information and current location which includes latitude and longitude to whom may concern. Moreover, any operation of the user should be easy to complete due to their difficulties resulted from visual impairments. Therefore, the structure of this project is basically clear, which consists of two parts - a smart blind stick providing function of navigation and emergency response and a mobile phone' s application monitoring location of the user and receiving signal from the stick. Communication between the smart stick and the mobile phone is achieved with Socket, where the Raspberry Pi on the stick works as the server and the phone works as the client. The scenario of communication is shown by Fig 1.

The reason that we choose Socket communication is that it allows the user to connect a simple client for testing by manually entering input and checking the corresponding response. Meanwhile, the communication can run on several machines, which is useful for scalability and is very helpful in debugging when working on embedded software.

When interacting with the application interface on mobile phone, the user needs to input a destination by voice command and triggered the transmission of signal by a simple action, like touching the screen. After receiving the routing information sent back, Raspberry Pi should process and issue command further to control related module. One of the most common method is notifying the user with tactile sense, which is suitable in our case. We connect four motors representing different directions (forward, back, left and right)to Raspberry Pi and place them on a circuit board fixing on the blind stick. In this way, the function of navigating can be achieved by informing the user with tactual command.

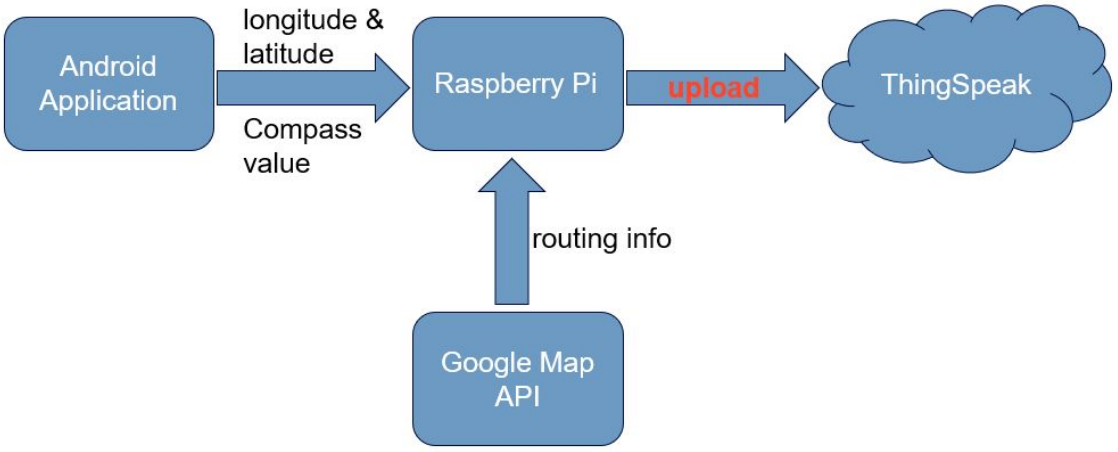


Figure 2 - Flow of information within architecture for Smart Sitick

Fig 2 shows the integral architecture of our project which involves all crucial parts - an Android application running on mobile phone which works as the Socket client, Raspberry Pi working as the Socket server which receives values or information sent from mobile phone or Google Map API and issues corresponding commands to integrated modules, and also a cloud platform call ThingSpeak which provides fundamental data service including caching and read/write. Since smart phone usually has the function of computing the current longitude & latitude and compass value based on the GPS module installed inside, Raspberry Pi can treat this data as the source information of navigating function after receiving from mobile phone. When obtaining the destination information from mobile phone, Google Map API can feedback relative routing information involving multiple legs on the routing road to Raspberry Pi. As a microprocessor, Raspberry Pi can process all those data and formulate various guidance command for each piece of the routing path. Furthermore, Raspberry Pi can upload all data to ThingSpeak, and a web page is used to display the current location of user after acquiring data from cloud. Meanwhile, if any emergency happens, doctor or family members of the user will be notified with signal and location information shown on the webpage.

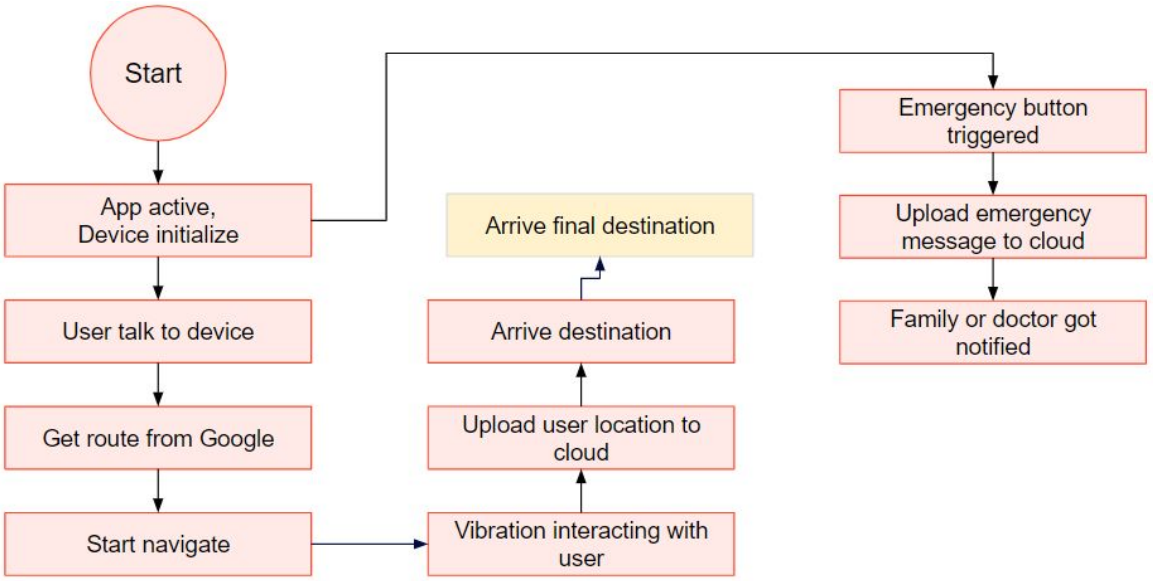


Fig 3 shows the logical algorithm of our project, which basically summarizes the whole processing flow. The start point of operation is inputting the destination of interest via voice command. After the procedure explained above, the user receives the tactile indication from vibrating of the motors and finally reaches the planned location.

Reference Paper Description

A lot of research on blind navigation such as [1], [2], [3] uses RFID for navigation. Those solution need a pre-setup of the whole moving range. In addition, considerably works such as [5], [6], [7] have done with ultrasound or ultrasonic sensor to avoid obstacles while walking. Yet here in our project, we aim to provide a supplementary service to users on top of white cane (stick) and even service dogs.

In paper [1], L. Ran et al. designed a wearable navigation system for blind pedestrians. They used Differential Global Positioning Systems (DGPS) to locate users in outdoor scenario and ultrasound positioning service for indoor scenario. Besides that, voice commanding and responding are included in their system. However, their system uses a built-in location database, and all paths are calculated using built-in location information. When navigating in a new place, it cannot play a good role. In our project, we used Google map API to give a real-time, internet powered navigation for users. Also, considering the using scenario, users already got a cane and sometimes they may have a navigation dog to help them avoid potential obstacles, our project doesn't include any ultrasound sensors, which could further reduce hardware budget.

In the work of R Velázquez et al. [2], the author proposed a navigating system with similar considerations of our project. They utilized smart phone as a backbone service, providing longitude, latitude, and orientation. The smartphone will then send those data to a cloud server and later forwarded to a remote computation station. The station connecting to OpenStreetMap will then generate the actual command and send it to the RF module on the microcontroller. The microcontroller enforces those command by vibrating the motors on the soles. This system is relatively complicated and costly than

ours. A remote laptop is required to play a role of remote station. And user need to move in the scope of the RF signal, which restricts the user's movement.

The author of paper [8] proposed a traffic lights detector for blind navigation. They implemented a system on top of smartphone, which could collect GPS data and get route command from the cloud service. TTS is then used to navigate visually impaired users. What need to mention is that they utilized machine learning to recognize images from camera of the smartphone. Giving the traffic light signal and transform it to voice for the user. The machine learning computation need considerable resource, also, they need to consider the timeliness of the voice commands: if they do all the computation on the cloud, the effects of the system depend on the latency of the wireless connection.

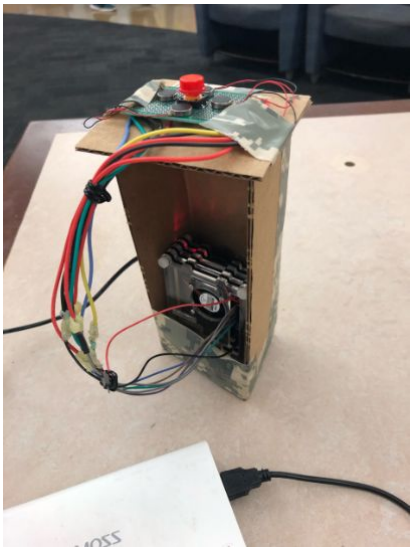
Also, paper [9] try to achieve navigation with visual signal from camera. The author developed a wearable solution to navigate users with a portable depth camera to extract information of surroundings. Their solution provides navigational commands through feedback from vibration motors embedded in a waistband. Contrary to above works, they developed a real-time navigation algorithm to trigger vibration of the motor. However, they didn't give a complete navigation solution. If the user needs to go to a specific place, this system is not helpful since it doesn't have location data and route data.

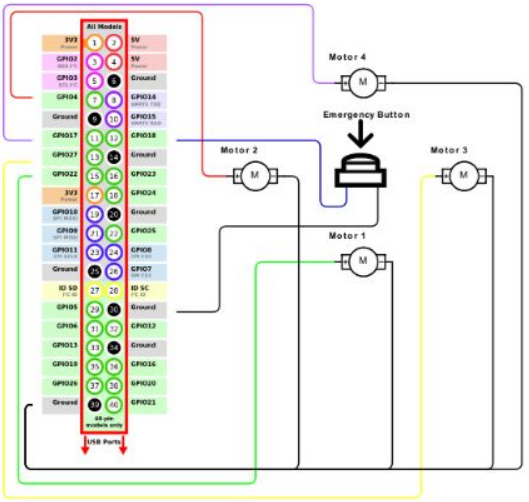
Reviews above shows that our proposal of navigating visually impaired users through vibration with smart phone is feasible and helpful. We put the generation of commands in microcontroller embedded in the stick. This could not only avoid the latency from the cloud computation and wireless connection but also reduce the cost of potential computation devices or cloud service. In addition to that, we used Google map api to give a real-time route. Since google keeps update their location data, the route we get should be optimal.

Hardware Experiment Description

For our project, we choose Raspberry Pi 3B+ as the microprocessor. At the start point, we faced two options - Raspberry Pi and Arduino Uno, which are also two main choice for most developers. However, Raspberry Pi is best used when you need a full-fledged computer: driving a more complicated robot, performing multiple tasks, doing intense calculations (as for Bitcoin or encryption)[reference: https://makezine.com/2015/12/04/admittedly-simplistic-guide-raspberry-pi-vs-arduino/]. After making the plan for basic functions and approaches, we noticed there exists possible extra functions we could add. Meanwhile, Raspberry Pi is appropriate for operating multiple programs at the same time. If we expect to achieve voice command

when the microcontroller processing received data from the client side, Raspberry Pi could be definitely qualified on this work.

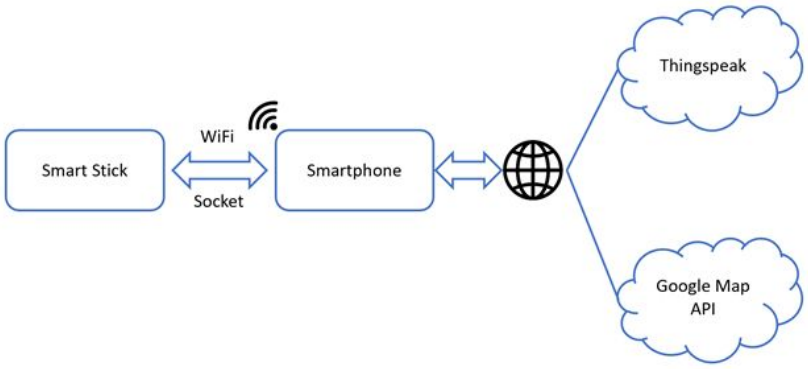




Two diagrams above (Fig 4 & 5) show the appearance and circuit structure of the hardware part, respectively. Since Raspberry Pi needs power supply from a USB cable, we use a portable battery connecting to it when testing our project. However, there is some unstabilizing factor as the highest current exported from the portable battery is 2.1A, while the standard current suitable for Raspberry Pi is 2.5A. In this case, the device can still work normally, but the working efficiency could be lower. Fig 5 presents the integrated modules and main board. Four Coin Mobile Phone Vibration Motors which needs power supply of 3V are displayed in four different directions on a panel sticking on the bottom of the device. The emergency button is displayed in the middle of panel. The user can trigger emergency signal by pressing the emergency button and waiting for approximately 2 seconds, and then all motors will keep vibrating and emergency information will come out on the webpage. As it shown on the schematic view, all modules are linked to different pin ports which can supply direct current, and GND port, respectively.

When we actually tested our hardware and measured its performance on campus, there exists some problems about the manifestation effect. It seems that it is hard for the user to distinguish which specific motor is vibrating since all modules are on the same panel. In this case, maybe a separate panel for different module could be a better option, although the cost of production could be risen.

A description of the network architecture



In order to provide a real-time, automatic wrong-walking correction, intelligent navigation service to ensure the correct path, our smart stick needs the support of the Internet. A lot of smart devices today are using WiFi to access internet, however, in our project, considering the user need to walk outdoor, we couldn't assume that WiFi is available along the route. Besides, we believe that cellular signal coverage is much better than public WiFis. As a result, we enable the "WiFi Hotspot" function of smartphone, generating a WiFi access point from smartphones cellular network and users don't have to subscribe another data plan for the smart stick. Also, we have considered using Bluetooth as communication interface between smartphone and smart stick. Yet, Bluetooth is not designed for real-time internet access and will increase the design complexity on both smart stick side and smartphone side, though Bluetooth is more battery friendly. Thus, based on WiFi hotspot, the smart stick could talk with smartphone, access Google map API and Thingspeak as well.

As for the communication between smartphone and smart stick, on top of WiFi, we used socket over TCP. Socket gives an abstraction of lower layer (layers that lower than Application layer in TCP/IP model) service to applications. After smart stick connected to the WiFi of smart phone, it will be allocated a private IP address. Then, the smartphone and the smart stick are under same private wireless network. We let the smart stick as the socket server and smartphone as the socket client since it's easier for smartphone to input the server's IP address. After established the socket connection, the smartphone and smart stick could send any bytes to each other. Here in our project, the smartphone keeps sending latitude, longitude and compass value to the smart stick every 0.5 seconds and smart stick will use them as need. This guarantees that smart stick always gets the updated location and orientation of the user.

In terms of modifications created over the course of the semester, we had both hardware and software adjustments that needed to be adjusted for our Smart Stick to function properly. At first we pursued coding everything in entirely python, but this was not efficient for working with smartphone applications. When deciding between working with iphones or androids, we began developing on an iphone and switched over since we had a spare old android phone that could work as a testing device for the Smart Stick. On the hardware side, we decided to simplify the vibration motors system since we were developing the proof of concept. Originally, the motors were going to be all in a circular fashion in which a hand could wrap around it. This was not needed to show the

basic idea of indicating the user where to go, so instead used only four motors that were attached to a cardboard box to emphasize the basic idea of the Smart Stick concept.

A large point of discussion within this project are which duties would be taken up by the Smart Stick itself vs the phone in terms of collecting data. At first we wanted the Smart Stick (Raspberry Pi) to do everything (collect GPS, compass value, voice input, etc) and for the phone to be used mainly as an app interface if needed and a source for cellular signal. However, for the purposes of a proof of concept it was much more efficient to extract GPS, compass values, and user input from the phone itself and to send it to the Smart Stick. This would trigger a series of Python coded algorithms of taking in the data and combining it with Google API to calculate the route info of where the user needed to navigate to. This avoided the Smart Stick from needing additional modules and lowered the overall budget of the project.

Another area that we modified our model was in terms of how we would navigate the user. At first we hoped to use simple vibration commands to indicate when to turn and hope the person would turn correctly. When the user ended up at the wrong location we created a plan to notify the user when they were outside a specified correct path. This system wasn't efficient since it served to correct the issue, instead of preventing the user from navigating the wrong way initially. Instead, we decided on an algorithm in which we continually notify the user at the correct direction. If the compass has an error, it should recalculate itself every update and give the next correct direction the user needs to navigate to.

The effectiveness of the new updates we added and the changes we made ensured that our product was more user friendly. The app allows for voice input for those that cannot type on a keyboard and we use Google API to lookup the closest location associated to what was said.

Conclusion

Our project can complete all basic functions that were within our initial plan at the beginning of the year. We also completed some extraneous goals that we had time to accomplish such as implementing the voice module, the emergency signal being sent through text message, and using Google Places API to autocomplete what location the user is looking for. However, the Raspberry Pi motors did not meet expectations for indicating the user direction. This is due to the motors being too close together and causing the entire device it is attached on the outside of to vibrate. In a industry setting the material of the device needs to insulate all vibrations except for the area of concern (each point of the stick where a vibration is expected). Outside research shows that there are other methods that achieve better functionality through the use of vibration motors on a waist belt, due to its larger shape and size. The size and shape plays a major role in how the vibrations are felt and how comfortable the device is to use.

Although in commercial use the Smart Stick should handle most (if not all) user interactions, we implemented the set up and data collection through the smartphone instead and programmed it to send the data to the Smart Stick to analyze and calculate route information. This functionality proved to be useful since data on smartphones usually provide very accurate results. We were able to use the results of the route information from Google Directions API to calculate if the user is facing toward or very close to their next checkpoint within their path to the destination. By following all checkpoints successfully, a blind user can navigate to any location successfully. The Smart Stick would serve as a supplemental device to the guide dogs often found with visually impaired users within commercial use.

Reference:

[1]. Ran, Lisa, Sumi Helal, and Steve Moore. "Drishti: an integrated indoor/outdoor blind navigation system and service." Pervasive Computing and Communications, 2004. PerCom 2004. Proceedings of the Second IEEE Annual Conference on. IEEE, 2004.

[2]. Velázquez, Ramiro, et al. "An Outdoor Navigation System for Blind Pedestrians Using GPS and Tactile-Foot Feedback." Applied Sciences 8.4 (2018): 578.

[3]. Ding, Bin, et al. "The research on blind navigation system based on RFID." Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on. IEEE, 2007.

[4]. Chumkamon, Sakmongkon, Peranitti Tuvaphanthaphiphat, and Phongsak Keeratiwintakorn. "A blind navigation system using RFID for indoor environments." Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference on. Vol. 2. IEEE, 2008.

[5]. Bousbia-Salah, Mounir, et al. "An ultrasonic navigation system for blind people." Signal Processing and Communications, 2007. ICSPC 2007. IEEE International Conference on. IEEE, 2007.

[6]. Bousbia-Salah, Mounir, Maamar Bettayeb, and Allal Larbi. "A navigation aid for blind people." Journal of Intelligent & Robotic Systems 64.3-4 (2011): 387-400.

[7]. Aguerrevere, Daniel, Maroof Choudhury, and Armando Barreto. "Portable 3D sound/sonar navigation system for blind individuals." 2nd LACCEI Int. Latin Amer. Caribbean Conf. Eng. Technol. Miami, FL. 2004.

[8]. Angin, Pelin, Bharat Bhargava, and Sumi Helal. "A mobile-cloud collaborative traffic lights detector for blind navigation." mobile data management (MDM), 2010 eleventh international conference on. IEEE, 2010.

[9]. Wang, Hsueh-Cheng, et al. "Enabling independent navigation for visually impaired people through a wearable vision-based feedback system." 2017 IEEE international conference on robotics and automation (ICRA). IEEE, 2017.