Smart Door Guardian

Taozhan Zhang, Chuwei Chen, Zhaozhong Qi
{zhangta6, chenchuw, zqi5}@bu.edu

1. Introduction

In this project, we plan to build a smart door guardian system that allows the user to enter the home without bringing the key. In the meantime, the system is mainly designed by enhancing the security of the home and integrating techniques of the networking and physical world between devices and devices. The smart guardian system uses real-time face recognition as a primary option, that identifies whether the person is the host or not. The system also supports the recognition of the voice by asking user to provide voice password through a microphone.

In a certain state, the system will not be able to sleep until the alarm signal has been triggered. Meanwhile, by pressing the button of single click, it will wake up the system, and initialize the services to allow the Recognition counter of the faces and listen to cipher without a specific order. Our Smart Door Guardian will be able to disparity the right standing point of the person, which by needing to check their cipher voice correctness or Counters that if and only if can be matched in our encoded face datasets in pre-recorded.

It is a comprehensive cloud computing and Internet of security things project. The primary purpose of this project is to use the techniques we learnt from the class, using secured ways of communication to integrate the separated functionality as an application by accessing the Ethernet.

2. System WorkFlow

Voice input and conduct certain actions such as informing the host or opening the door. Specifically, it will follow the instructions as shown.

- One-click start: Users can decide one of the two measures to check its identity, either by face recognition or voice password, by single or double clicking the IoT button.
- Real-time face recognition: Recognize the incoming person's identity by face recognition. Triggered by single-clicking the IoT button, the program will start running real-time face recognition and open the door if the person is the real host.

- Voice password recognition: Triggered by double-clicking the IoT button. The program will recognize the incoming person's identity by checking if the person has given the pre-set voice password.
- Sentiment analysis: Depending on the user's voice input, we obtain the sentiment score to decide to open the door or not. Aim to prevent robbery.

The door system has the following states: Non-Start, Sleep and Alert. When Sleep, it does nothing until someone presses the IoT button to save power. In alert state, it will immediately inform the host to check the camera until the host deactivates the alert signal. The workflow of the door system is shown in figure 1.

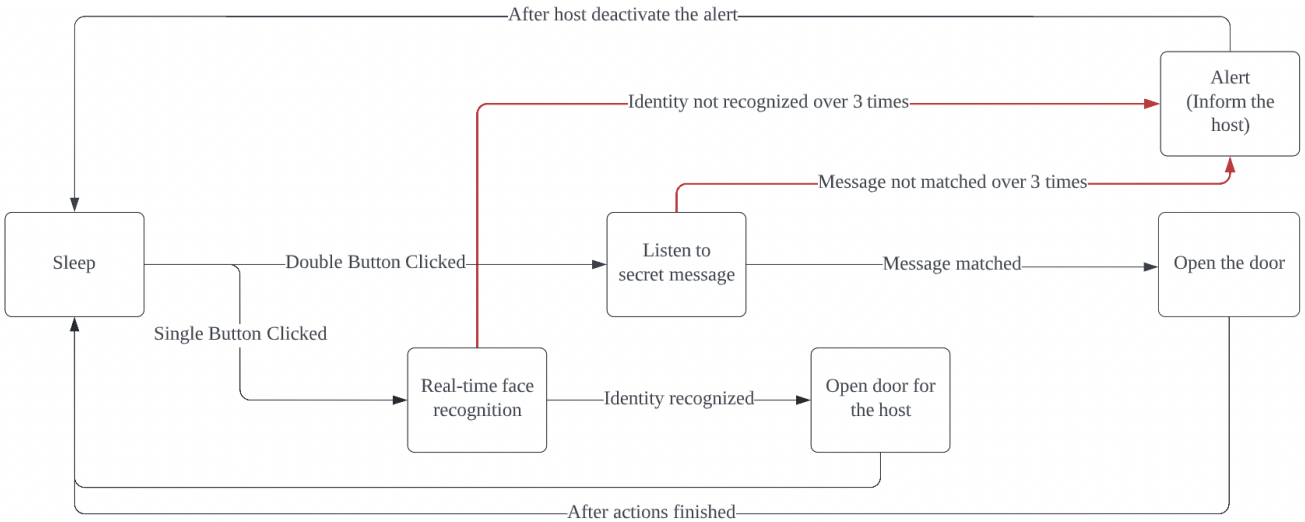

Figure 1. Smart Door Guardian System Diagram

3. Technical Approach

Four different modules are included in this project, technical approaches are discussed below

Face Recognition:

The main functionality of the face recognition module is identifying the targets as the host or the strangers. and let the system decide whether to open the door for the visitors or send out the alert signal. Our face recognition integrates the technique by using OpenCV, Python, and deep learning to build an encoded face dataset. As far as you might be not familiar with the process of the Face-Recognition, we will introduce the simplified theorem here.

The main theory and the implementation can be separated into two steps. Step 1, which will be face-encoding. Instead of outputting a single label (or even the coordinates/bounding box of objects in an image), we are used our encoding-faces algorithm to output a real-valued feature vector as comparison which calculated by the model.

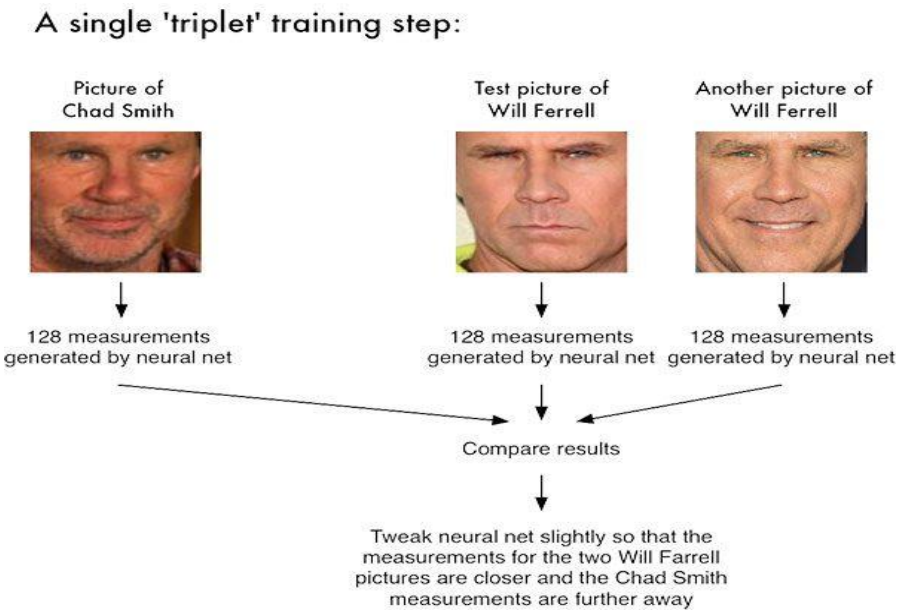

Figure2. Face-encoding theorem, takes in a single 'triplet' faces rectangle as input, and output a 128 dimensional encoded feature map of the face representation.

The process and technique which could be shown as the above,

Figure3. Face-encoding in details, takes in a single 'triplet' faces rectangle as input, and output a 128 dimensional encoded feature map of the face representation.

For the dlib facial recognition network, the output feature vector is 128-d (i.e., a list of 128 real-valued numbers) that is used to quantify the face. Training the network is done using triplets, quantifying the faces in our training set. Keep in mind that we are not actually training a network here — the network has already been trained to create 128-d embeddings on a dataset of ~3 million images.

Figure 4. Sample Face-Recognition Dataset

Since face recognition is tried to first detect the faces in the photo or the profile included in each frame of the video. We used the technique of the OpenCV to

Voice Recognition:

The voice recognition module can clearly figure out what the visitor says, analyze the emotional signal, as well as an additional authentication function to require the visitors speak out the passwords, if the recognition results do not fit, the system will send out the alert to the host.

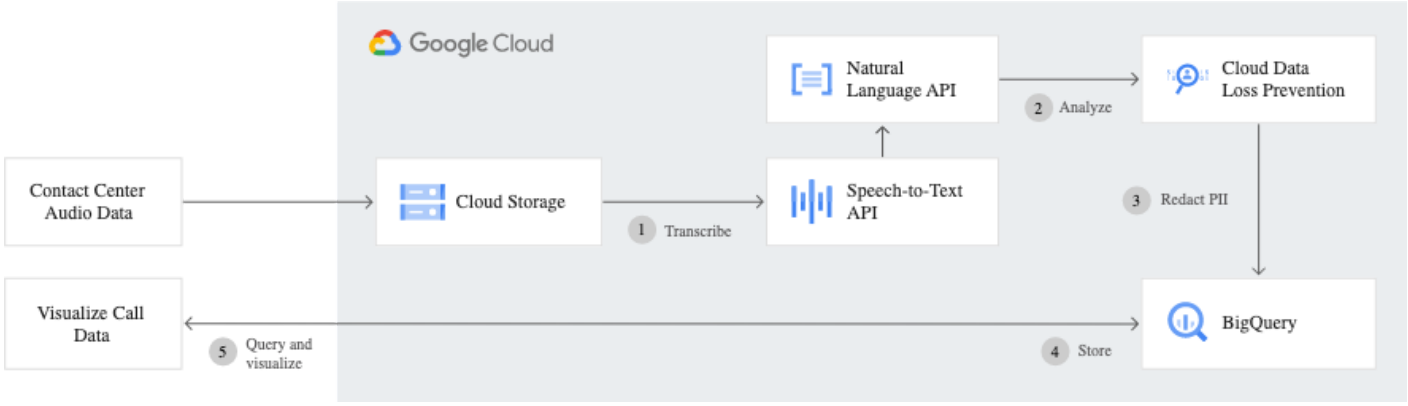

Figure 5. Workflow of Google's Speech-to-Text API

We accessed Google's service Speech-to-Text API as shown in figure 5, which is a pre-trained Machine Learning model that can convert one's voice (.wav file) input to an editable, verbatim transcript. Along with the Raspberry Pi and a microphone, we could capture the visitor's voice input and convert it into a transcript, then check if it matches the pre-set voice password (pre-set voice password: "networking the physical world"). Such a voice recognition model is highly reliable with an average conversion confident of 95.28%, meaning that the probability that voice input could be clearly, completely converted is over 95.28%.

With such an appreciable success rate, we can avoid the possibility of the owner accidentally saying the wrong password and not being able to open the door.

AWS IoT connection:

Every devices in this project is connected through Amazon's IoT cloud service. By registering each device, we obtain the certificate, private key, and root certificate for each device. We could then use these certificates to let the devices subscribe to the sample MQTT topic to communicate with each other. We choose MQTT as our transmission protocol because it is highly lightweight and very suitable for small, computation ability constrained devices such as raspberry pi.

M2M communication:

In this project, we implemented M2M communication between three end-point devices: Seeed IoT button, Raspberry Pi (Client-1) and Raspberry Pi (Client-2). By using the same MQTT topic as our channel, we build wireless machine to machine communication: IoT button → Raspberry Pi → Door.

4. Technical Difficulties

- Initially, we decided to use the FRDM-K64F board as our door handler by subscribing to the MQTT topic and open or close the door according to the incoming message. However, the configuration of FRDM-K64F to the AWS IoT turned out to be a huge problem for us. Despite the confusing user guideline provided by Amazon, the complexity of the example demo from MCXpresso is largely underestimated, with hundreds of header files full of static structure code, it is a huge engineering task to customize a program based on these codes. We finally have to replace the FRDM board with another Raspberry Pi to be our door hander.

5. Discussions on the technical successes of the project

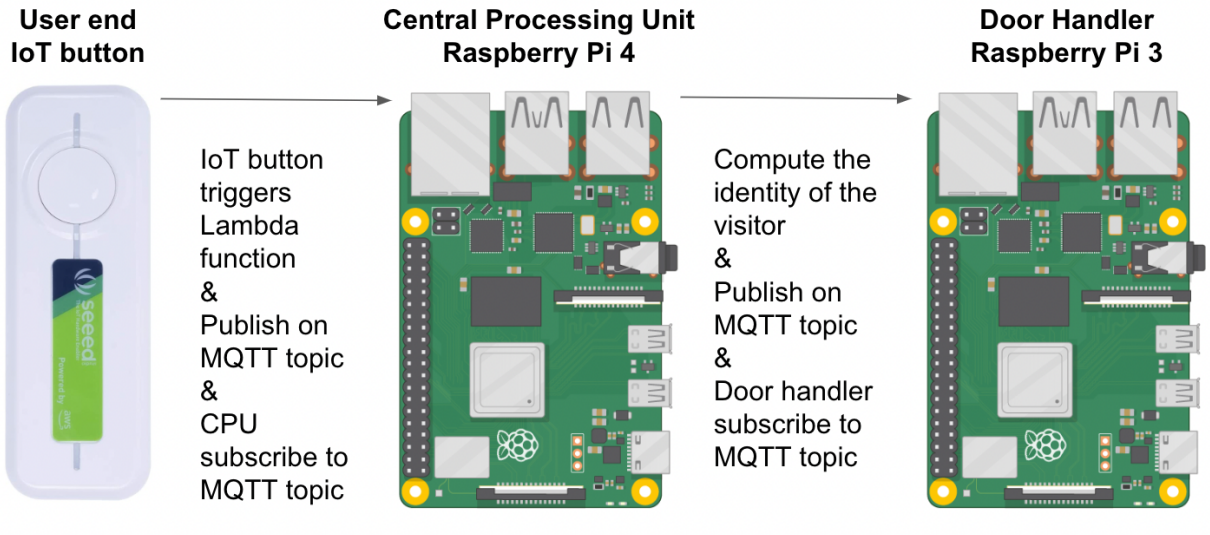

Figure 6. Network architecture of interconnected things

- In the project, we successfully built the modules of face recognition and voice recognition as shown in figure 6. Face recognition and voice recognition modules are integrated on the raspberry pi 4, and achieved the communication between it and Raspberry Pi 3, as well as remote control via AWS IoT button.
- As the results of calling strong open source APIs, both the face and voice recognitions with high accuracy, as well as extremely fast learning efficiency. The face recognition module can identify the targets as the host or the strangers within 2 seconds, and let the system decide whether to open the door for the visitors.
- The voice recognition module can clearly figure out what the visitor says with an accuracy of over 95 %, as well as an additional authentication function to let the visitors proving their identity.
- Within the implementation of AWS IoT MQTT, the whole processing and reacting period can be less than 10 seconds(the actual time depending on the internet service quality and the raspberry pi status), while the demand of security is fulfilled as every devices on the IoT network needs independent, unique certificate, policy, and private key to configure.
- During the project, we learned how the MQTT works, as well as the implementation of the APIs and the powerful functions on various tools , like the lambda function compatible for various languages and platforms on AWS IoT, as well as extract the detailed key information for the generation of credentials. Also, we figure out how to deal with several firmware update problems when those provided by the official website are not functional.

References

[1] Google Cloud Natural Language: “Speech-to-text basics | cloud speech-to-text documentation | google cloud,” *Google*. [Online]. Available: https://cloud.google.com/speech-to-text/docs/basics. [Accessed: 05-May-2022].

[2] TensorFlow: “Tensorflow,” *TensorFlow*. [Online]. Available: https://www.tensorflow.org/. [Accessed: 05-May-2022].

[3] OpenCV: “Deep Neural Network module,” *OpenCV*. [Online]. Available: https://docs.opencv.org/3.4/d6/d0f/group__dnn.html. [Accessed: 05-May-2022].

[4] Amazon AWS: “Whitepapers,” *Amazon*. [Online]. Available: https://docs.aws.amazon.com/whitepapers/latest/aws-overview/what-is-cloud-computing.html. [Accessed: 05-May-2022].

[5] Seeed IoT button: B. Zuo, “Seeed IOT button for AWS,” *seeedstudio*. [Online]. Available: https://wiki.seeedstudio.com/SEEED-IOT-BUTTON-FOR-AWS/. [Accessed: 05-May-2022].