Rosie CS 4400 Project

Geunyoung Choi and Linh Hoang

***Abstract* - Security is a big concern for many Americans, yet more than 1 in 5 people leave their doors unlocked [1, 2]. The main reason is the hassle of physically locking and unlocking the door every time. With the emergence of the Internet of Things, our approach was to create a smart lock device which we called *Rosie*. Rosie has the capability to unlock the user's door with only the presence of a face using its face recognition. Rosie also has a voice interface which allows non authorized users such as visitors to talk with Rosie and communicate with the owner. Throughout our development, we did encounter a couple incidences with the facial recognition due to storing only facial vector and the quality of the camera used. In this paper, we describe our findings and the capability of Rosie.**

1. Introduction

The Internet of Things has become a reality with more accessible and cheaper components. This has allowed a lot of our day-to-day appliances that we use today to be incorporated with modern technologies. There are smart devices in nearly every home that are interconnected in the Internet of Things such as smart light bulbs, home assistants like Google Home, and smart locks [3, 5]. These devices are integrated to help us in everyday tasks and activities while also fitting seamlessly and natural in our world.

Nearly all smart locks in the market currently will require an application on the user's phone to control the lock on the door and does not include a voice interface which prevents any communication with visitors [5]. There are also a few smart locks with a built in camera on the market currently; however, none of them has the capability of facial recognition.

When we decided to create our device, Rosie, our goal was to ensure security while also playing a seamless role in people's lives. Our idea was to create a smart lock with a voice interface that would seamlessly unlock the door when an authorized user walks to the door automatically without the need of a key or a phone. On the other hand, when a visitor approaches the door, they will be able to communicate with Rosie to notify the owner.

2. Design

2.1 Hardware Components

To create Rosie, we used NVIDIA's Jetson Nano, a camera, a microphone, and a speaker. The NVIDIA Jetson Nano is an embedded system-on-module (SoM) and developer kit from the NVIDIA Jetson family, including an integrated 128-core Maxwell GPU, quad-core ARM A57 64-bit CPU, 4GB LPDDR4 memory, along with support for MIPI CSI-2, PCIe Gen2 high-speed I/O and 4 USB ports[4].. The camera we used is the 8MP IMX219-77 Camera. The microphone we use is an Fifine USB Microphone and the speaker is an Anker portable speaker. (check the camera)

2.2 Workflow

Our smart lock works like the following:

1. The visitors come in front of the door, and press the button to activate the Google Assistant Action smart lock.
2. The smart lock detects if there is a known person inside its field of view and offer to open the door if there is
3. If the smart lock does not detect an owner in its field, it offers the visitors to contact the point of contacts specified by the database

2.3 Architecture

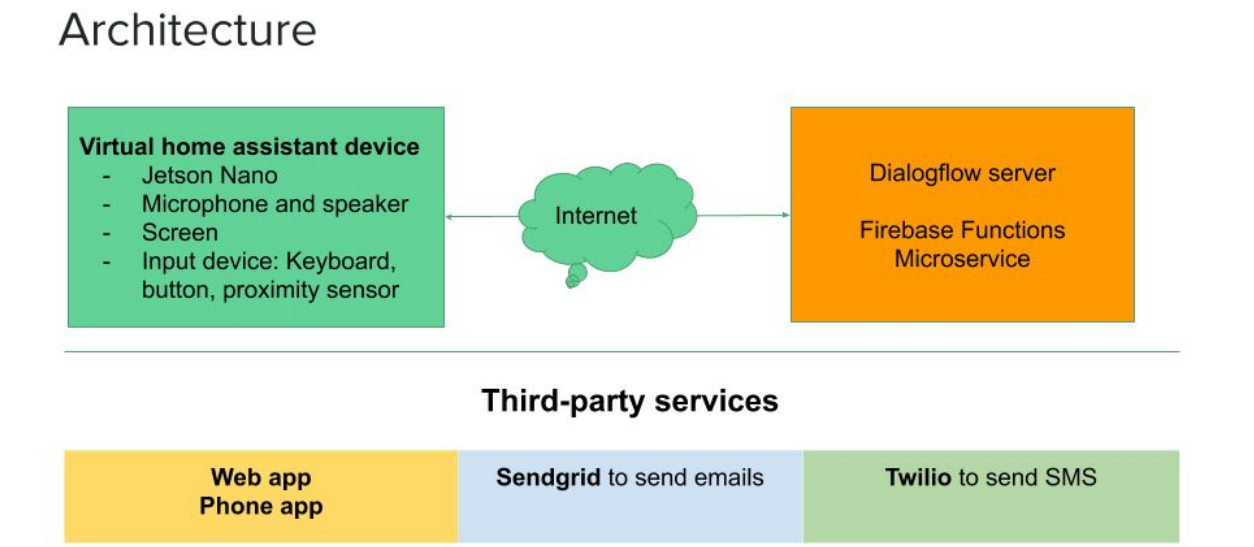

Figure 1: The Architecture of Rosie

Our smartdoor bell comprises of an offline component built using the Jetson Nano, and a cloud component. The architecture of Rosie is shown in Figure 1. The offline component has a software to do facial authentication. Our authentication software provides the context of the people who are currently present at the door to a python script that runs a voice user interface on the Jetson Nano. The framework we use to develop our voice user interface requires the Jetson Nano to have an Internet connection and a backend on the cloud. We set up our backend to run our voice user interface framework and interact with third-party services. We detail our software components in the next section.

2.2 Software Components

Facial Recognition

Our facial recognition software is built in python using Dlib, OpenCV and Gstreamer. Jetson Nano supports hardware decoding from the camera using the **gstreamer** software. We set up the decode software to capture the frame at 1280 x 720 pixel for 120 frames per second and feed it to the **dlib** facial recognition software. The raw stream is preprocessed to be 0.25 times smaller before it is fed to **dlib**. As there is a dedicated hardware decoder, dlib can fetch frame on demand to execute facial recognition and facial features extraction.

We use the **dlib** face recognition python toolkit [6] for our facial recognition capability. The **dlib** toolkit maps the images of a human face to a 128 dimensional vector space to distinguish one face from another. This facial feature can be stored offline and associated with an identity for authentication purpose [11].

The facial recognition software also needs OpenCV to render the frame and face boxes for debugging purposes and to preprocess and transform the frame to the facial recognition software

Google Assistant SDK:

Our smart lock use a Google Assistant Action as the voice interface. Google Assistant Action is a framework to extend your own applications on top of the Google Assistant technology - the natural voice user interface virtual assistant built by Google [7]. Google Assistant SDK allows developers to put the control of Google Assistant on a customized device [8]. We use this SDK to configure our smart locks to initialize our Google Assistant Action **Rosie** and authenticate our smart lock device with a unique **ID** when users start the smart lock program. We also use this SDK to send the context of the door to the backend server for the processing of relevant Google Assistant requests.

DialogFlow:

To build our Google Assistant Action **Rosie**, we use DialogFlow. DialogFlow is a framework to build a conversational flow; it supports entities extractions from the input of the user's request and provides an node.js API to build customized conversation flow, as well as to build a connection to a third-party service [9]. Third-party services can be a web app, a phone app, a communication service such as **Sendgrid** and **Twillio**. We use the Dialogflow customized backend API to connect to a virtualized lock as a demonstration for its third party connection capability.

D-Bus:

We use D-Bus as an IPC mechanism between the facial authentication software and the Google Assistant Embedded SDK script. The facial authentication software uses D-Bus to communicate with Google Assistant Embedded SDK script who are currently present at the door. This information is then sent to the DialogFlow backend server to serve as a context for the processing the conversational flow.

Firebase:

Firebase is is a mobile and web application development platform by Google [12]. We use the Firebase Cloud Firestore [14], a real-time database, to associate the device **unique ID** with the information of the owners of the device.

2.3 Testing Components

For testing purposes, we used a keyboard and a monitor to view Rosie's camera view. Additionally, we haven't created a physical door locking mechanism yet. Instead,

we used a virtual door that checks the status of the door stored in the firebase to make sure the locking and unlocking procedure was working correctly. We also created a web application that allows the user to check if the status of the door and also allows the user to remotely lock or unlock the door. This is created by the Firebase Hosting [13], a web hosting service and Firebase Cloud Firestore [14], a real-time database.

3. Evaluation:

We evaluated our facial features extraction and face recognition ability by measuring the performance of the program in frames per second. The facial recognition experiments are done under four different conditions: when either one, two, three or four faces are within the field of view of the camera. For each condition, we measured the performance of the program three times and stopped after the facial recognition software has completed 55 loops of facial recognition and facial extraction. Each condition's result are shown in the figure below.

Figure 2 shows that the stable frame rate is achieved after 20 loops for all four conditions. Surprisingly, with one face in the frame, for the first 20 frames, the frame rate can be as high as 6 frames per seconds and then the fps decreases to about 5 fps after. For all conditions, the initialization cost is quite significant, as we can see all the frame rates at the initialization are significantly lower compare to the stable framerate. There is also more drop in frame rate in the first 20 frames comparing to after 20 frames.

Figure 3 shows the average frame rate for each condition. With each additional face in the camera, the frame rate drop about 0.5. If 1 frame per second is the minimum performance for our smart lock than there should not be more than 8 people in the frame.

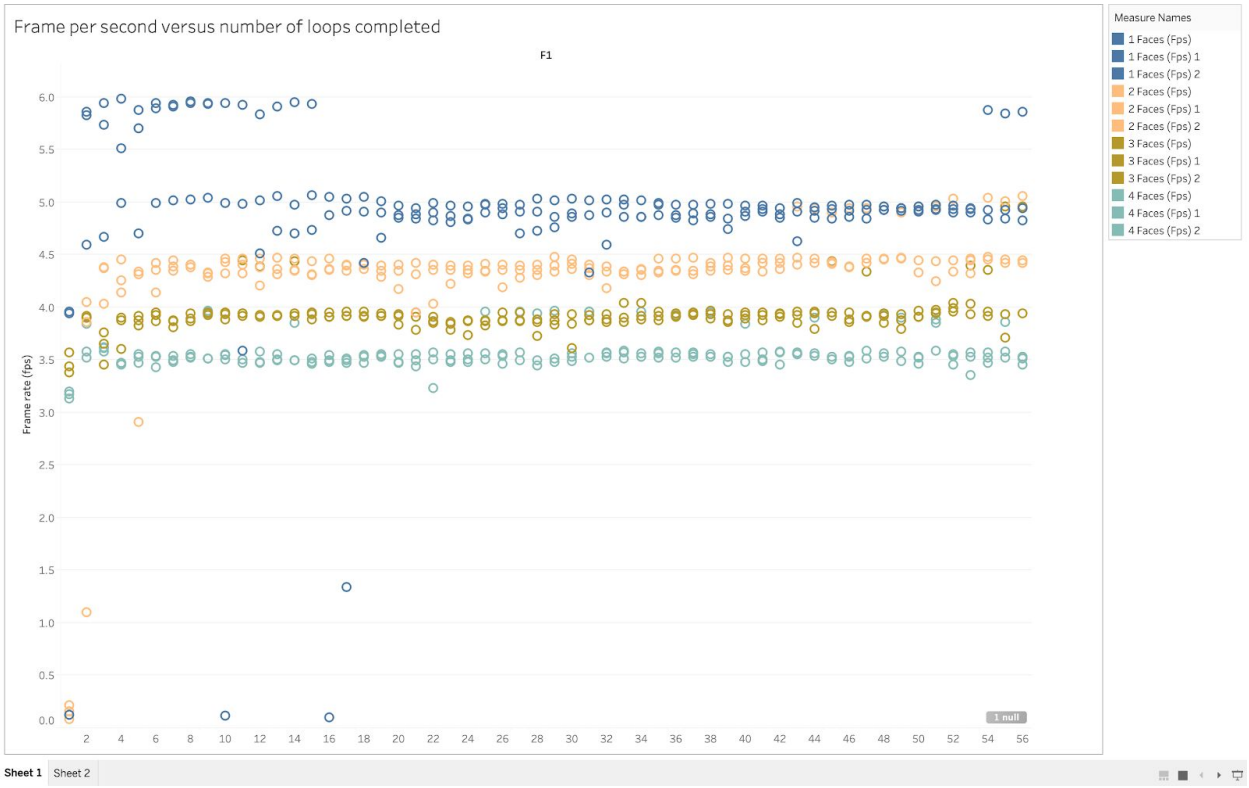

Figure 2: The figure shows the frame rate at each condition versus the number of facial recognition and feature extractions loops completed

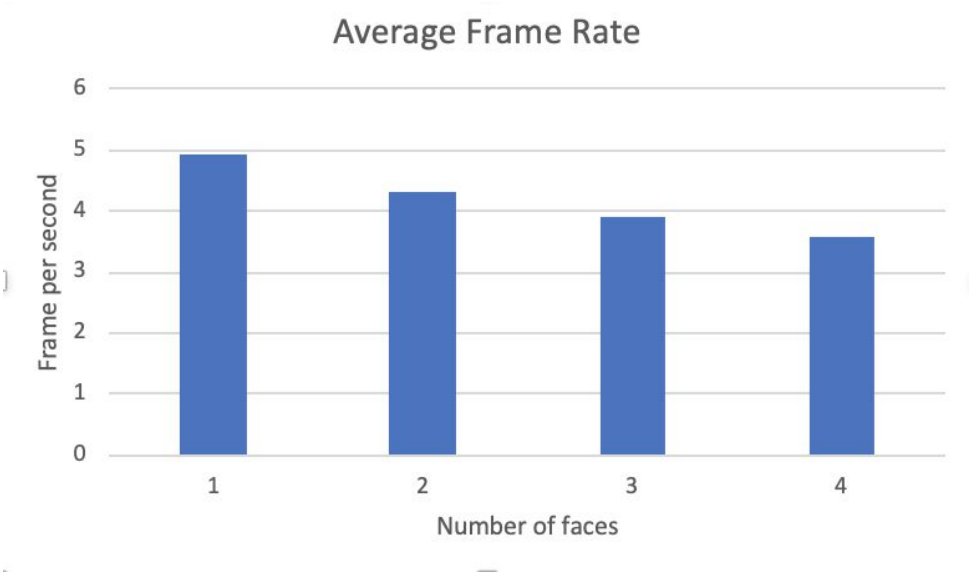

Figure 3: The figure shows the average frame rate versus the number of faces in the camera frame

4. Challenges:

We had encountered a few problems during testing of the accuracy of the facial recognition. There was an incident when Rosie was recognizing a Korean user's face as a different Korean's face. There was also an incident when Rosie wasn't recognizing an authorized user and thought the user wasn't authorized. The problems exist because of two issues, the quality of the camera and capturing of only one facial vector.

Rosie use a 320 x 180 size frame for facial recognition. In the future, with more powerful hardware, we can increase the resolution of the frame processed by Rosie. Rosie also use only one facial vectors for each owner face, which means that it still make many mistakes when identifying people. We can improve the facial authentication by storing multiple facial vectors of the owner's face by capturing multiple snapshots of different angle of the face for a better accuracy of the facial recognition capability.

For the prototype, Rosie's camera was always on and we had to physically press a button for Rosie to check the user's face. We intend on using an infrared sensor that would only turn Rosie's camera on when it detects someone is close by to detect their face.

5. Related Works:

There are similar products out there on the market currently. One product that has face recognition capability is Google's Nest Hub Max. The Nest Hub Max has a feature called Face Match which allows the system to recognize the face and accommodate the contents and actions such as sending messages or notifications suited for the user. However, the Nest Hub Max is a home assistant device and is not suited for controlling the lock on your door.

There are also few smart locks on the market such as the August Smart Lock or the SimpliSafe Smart Lock. Hilton Hotels also introduced smart locks in their hotels back in 2015 and allows guests to download their app to access their Digital Key which allows their phones to unlock the door with their phones. Most of the other smart locks on the market currently such as the August Smart Locks and the SimpliSafe Smart Locks have the same feature where the owners of the house can control their locks with their phones.

The biggest difference between the other smart locks on the market and our Rosie is the facial recognition capability and the voice interface. Rosie's facial recognition capability is almost instantaneous. Our test shows that Rosie can correctly distinguish a person's face at 2.4 meters away from the camera at around 5 frame rates per second which is faster and doesn't require any work. Another feature Rosie has is a voice interface. The voice interface allows everyone including visitors who won't have

the app communicate with the system and notify the owner. So if a delivery man had come to drop off a package, he can notify the owner of the house that a package had arrived. On the other hand, if a recognized friend or a neighbor had come to visit, they can be notified that the owners aren't at the house at the time.

6. Conclusion/Future Work:

The prototype that we have created serves a good idea of what Rosie can be. According to our evaluation, we didn't have any issue with the facial recognition's latency. However, we did encounter some issues when the system was not recognizing a face or recognizing a similar face as a different face. As a countermeasure, we plan on using multiple facial vectors with different angles of the face to handle this issue.

Additionally, we will need to add the physical mechanism to lock and unlock the door. Additionally, we created a web application for the user to be able to control the lock settings remotely on the phone; however, we plan on making an application for the owners with more functionalities such as getting a notification with a message from someone at the door, getting a snapshot of them with Rosie's camera, and a User Interface to manage owners' and admins' permissions.

7. References:

[1] "Locking Your Doors." *SafeHome.org*, 28 Nov. 2019, www.safehome.org/resources/locking-your-doors/.

[2] Wadler, Joyce. "The No Lock People." *The New York Times*, 13 Jan. 2010, www.nytimes.com/2010/01/14/garden/14nolock.html.

[3] Anstee, Darren. "Rise of the Internet of Things (IoT)." *TechRadar*, TechRadar Pro, 21 Nov. 2019, www.techradar.com/news/rise-of-the-internet-of-things-iot.

[4] "Jetson Nano Developer Kit." NVIDIA Developer, 19 Nov. 2019, developer.nvidia.com/embedded/jetson-nano-developer-kit.

[5] Looper, Christian de, and Monica Chin. "The Best Smart Locks." *Business Insider Singapore*, 13 Nov. 2019, www.businessinsider.sg/best-smart-lock/.

[6] Ageitgey. "Ageitgey/face_recognition." GitHub, 13 Nov. 2019, github.com/ageitgey/face_recognition.

[7] "Actions SDK Basics | Conversational Actions | Google Developers." Google, Google, developers.google.com/assistant/actions/actions-sdk.

[8] "Introduction to the Google Assistant Service | Google Assistant SDK." Google, Google, developers.google.com/assistant/sdk/guides/service/python.

[9] "Dialogflow." Dialogflow, dialogflow.com/.

[10] “Python: Python Bindings for D-Bus¶.” Dbus, dbus.freedesktop.org/doc/dbus-python/index.html.
[11] Lee, Frank. “A Practical Guide To Using Face Technology (Part I).” *IoT For All*, 5 Aug. 2018, www.iotforall.com/practical-guide-face-technology-part-1/.
[12] “Documentation | Firebase.” Google, Google, firebase.google.com/docs.
[13] “Firebase Hosting | Firebase.” Google, Google, firebase.google.com/docs/hosting/.
[14] “Cloud Firestore | Firebase.” Google, Google, firebase.google.com/docs/firestore.