Sullivan: Smart Door
Kyuha Jung, Minki Choi, Wonjun Choi
Smart Things: IoT
Documentation Created: Sept. 23, 2019
Last edited: Dec. 19, 2019

Goal: Our goal is to recognize registered users' faces to unlock the door using AI in open source computer vision library called OpenCV, with a Raspberry Pi computing system.

Reasoning: Door lock/security system has evolved as a means of providing both more convenient and secure life; from common and physical keys to PIN locks and biometric door locks. Our team began to wonder how effective it would be if we could implement more features. Less is more, but if the feature could contribute to peoples' safety and convenience at the same time, we thought it was worth it.

Our project is mainly aimed for children and elderly, but it is not limited to them so all will profit.

Possible Concerns: Our project makes use of a camera; implementing a camera into home IoT system may arise privacy concerns. However, that is not an issue in our case since the camera will be installed outside of the front door, facing outwards at all times.

Future reference: Our idea could further improve peoples' lives.
When a natural disaster happens or fire accident occurs next door, would young children know the procedures to stay safe, would the elderly be active enough to stay safe? According to the Korean Statistics Information Department, a lot of minors die from such accidents. We thought this was a huge problem because in Korea, many elders live alone in illegal buildings where they do not have the adequate fire extinguishing ability.

We thought this idea was something that not only the elders in Korea could benefit from but could target a much wider range of people. For example, the door with facial recognition ability will keep track of who went in and who came out in real time. When there is a dangerous situation, the information the door recorded, such as number of people left in the house, will be provided to the control center, ultimately sharing information with the police officers or firefighters. By doing so, they will have high efficiency, not having to break into every single home that are empty.

Below is a documentation for preparing the Raspberry Pi with OpenCV (Computer Vision). It is a tutorial made for Todd Holoubek's class, Smart Things: IoT, to guide on how to install OpenCV 4 on Raspberry Pi 3 B+ with Raspbian Buster as OS.

(We will refer to the "Raspberry Pi 3 B+" the RPi in short, in this guide)
The order of the documentation is as follows:

1. **Cleaning and Preparation of RPi (after installation of OS)**
   - 1-1 Power Management Settings
   - 1-2 Storage Expansion
   - 1-3 Removing Useless Programs

2. **Basic Settings: for camera module to communicate properly**
   - 2-1 Granting access to the camera module
   - 2-2 Configurations to enable the camera module

3. **Installation of Required Libraries to Operate OpenCV**
   - Refer to the separately attached document
   - 3-1 Update the System
   - 3-2 Install Dependencies and Libraries
   - 3-3 Setup a Virtual Environment
   - 3-4 Download and Install OpenCV to the Virtual Environment
   - 3-5 Test of Proper Installation

4. **Coding in python 3 (Facial Detection and Recognition)**
   - Refer to the separately attached documents
   - 4-1 Building Datasets (of Users to Unlock)
   - 4-2 Use the Built Datasets to Encode the User Data
   - 4-3 Control what RPi will do with Recognized User

5. **Coding Arduino and Connecting with Electro Magnetic Lock System**
   - Refer to the separately attached document

6. **References**
   - Representative References

1 Cleaning and Preparation of RPi (after installation of OS)

1-1 Power Management Settings

NOTE: The installation on the RPi takes a few hours of preparation, building and compiling. To prevent installation from aborting or halting issues in prior, make sure auto sleep or auto poweroff is disabled under power management settings.

1-2,3 Storage Expansion / Removing Useless Programs

Once you fresh install the official OS, the default user ID and password is: (pi/raspberry).

For this project, it is essential to acquire as much as storage possible. To do this, we remove unwanted software such as photo editor; software such as photo editor, office package is not necessary.

2 Basic Settings: for camera module to communicate properly

2-1 Granting access to the camera module

Secondly, our project will be using video camera module available at:

https://www.raspberrypi.org/products/camera-module-v2/

We need to grant the user, pi, access to the video; on the terminal, enter:

```
sudo usermod -a -G video pi
```

2-2 Configurations to enable the camera module

Add

```
start_x=1
gpu_mem=128
```

to the end of [/boot/config.txt]. A reboot is required after these steps. Done!

3 Installation of Required Libraries to Operate OpenCV

For step 3, refer to the additionally attached document named:

```
3-1_rpi_opencv_installation.sh
```

4 Coding in python 3 (Facial Detection and Recognition)

On desktop of RPi, create a new folder for your project.

From the installation directory, locate to [/opencv/data/haarcascades/]; copy the file:

```
haarcascade_frontalface_default.xml
```

paste the copied .xml file into our project folder.

Download the 3 python files from Google Drive and place them into the new project folder.

Run the following commands (in virtual environment, in the project folder directory)

4-1 Build the dataset using your faces

For step 4-1, refer to the additionally attached document named:

```
4-1_build_face_dataset.py
```

Execute the following by:

`python3 4-1_build_face_dataset.py --cascade haarcascade_frontalface_default.xml --output dataset/"USERPREFERENCE"`

Replace "USERPREFERNCE" with your name

Ex) I used WC for my initials Wonjun Choi

Once the code loads, a green box will appear around the detected faces. To capture the shot, press ‘k’ so save into the folder named “USERPREFERENCE”.

4-2 Use the Built Datasets to Encode the User Data

For step 4-2, refer to the additionally attached document named:

`4-2_encode_faces.py`

Execute the following by:

`python3 4-2_encode_faces.py --dataset dataset --encodings encodings.pickle --detection-method hog`

This python code will encode all the users’ facial information, according to the folder name you set at 4-1

4-3 Control what RPi will do with Recognized User

For step 4-3, refer to the additionally attached document named:

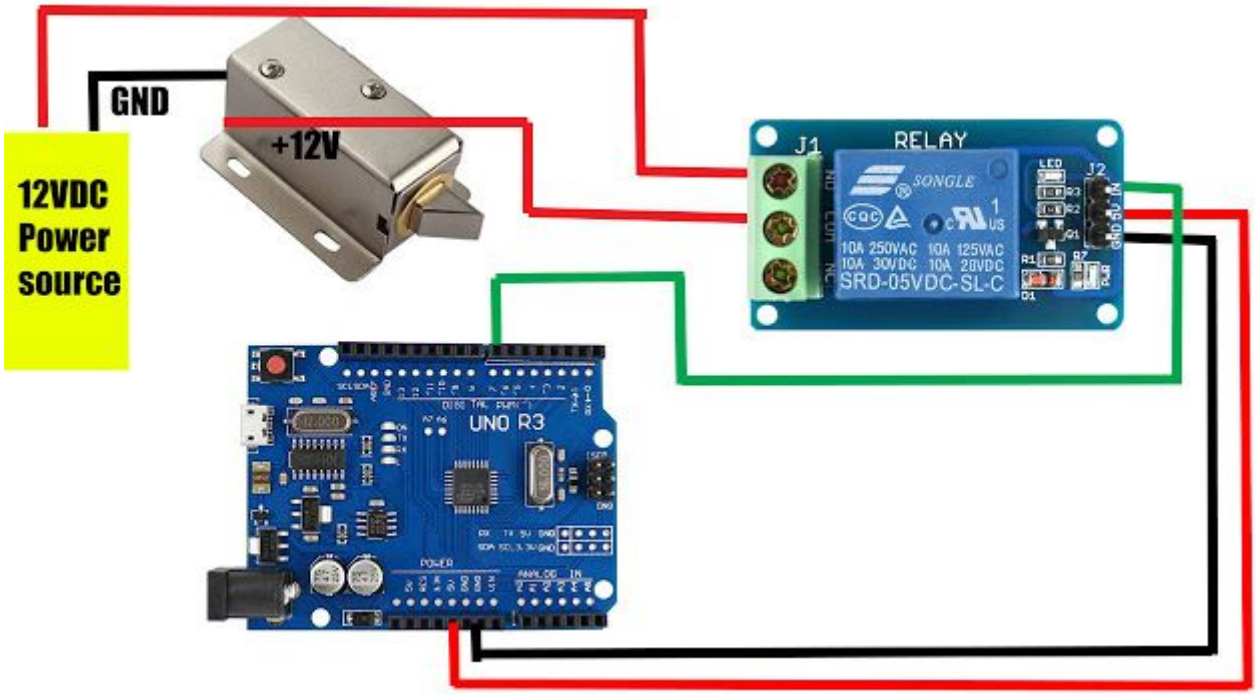
`4-3_pi_face_recognition.py`

Execute the following by:

`python3 4-3_pi_face_recognition.py --cascade haarcascade_frontalface_default.xml --encodings encodings.pickle`

In this code, I set it so that if my face (“WC”) is recognized for 3 frames, send character “y” over the serial connection to Arudino

5 Coding Arduino and Connecting with Electro Magnetic Lock System



This is the circuit wiring.

For Arduino coding, refer to the additionally attached document named:

`5-1_solenoid.ino`

6 References

Install OpenCV 4 on Raspberry pi 4 and Raspbian Buster
https://www.pyimagesearch.com/2019/09/16/install-opencv-4-on-raspberry-pi-4-and-raspbian-buster/

Install OpenCV 4.1 on Raspberry Pi 4 – Q-engineering
https://qengineering.eu/install-opencv-4.1-on-raspberry-pi-4.html

How to Connect and Interface a Raspberry Pi With an Arduino
https://maker.pro/raspberry-pi/tutorial/how-to-connect-and-interface-raspberry-pi-with-arduino

Arduino and Raspberry Pi Serial Communication
http://www.notforme.kr/archives/756

Relay Module + solenoid Door lock How to control them with an Arduino
https://www.arduinotutorialonline.com/2018/05/relay-module-solenoid-door-lock-how-to.html

Any questions, reach Wonjun Choi at jwchoi09@yonsei.ac.kr