

INTEGRATION OF SMART DOOR LOCK WITH FACE RECOGNITION BASED ON RASPBERRY PI 3 WITH GOOGLE ASSISTANT FEATURES

Ivan Surya Hutomo¹, Handy Wicaksono²

^{1,2}Department of Electrical Engineering, Petra Christian University, Indonesia

Article Info

Article history:

Received Jun 9, 2016

Revised Nov 20, 2016

Accepted Dec 11, 2016

Keyword:

Face recognition

Google Assistant

Digital assistant

Smart door lock

Raspberry Pi 3

ABSTRACT

This thesis will be focused on the integration of smart door with face recognition and Google Assistant. The smart door will be unlocked if face already recognized but the smart door will stay locked if the face is not recognized and the system will send an email notification to the house owner. The system will be using Raspberry Pi 3 as main microcontroller and servo as locking door actuator. The program will be developed using node-red, Blynk and MQTT platform which are very powerful for developing Internet of Things devices. All of the programs will be coded using Python language that commonly used in Raspbian OS for Raspberry Pi 3. Face detection method will be using Haar Cascade features and face recognition method will using Local Binary Pattern Histogram. Google Assistant integration will use Dialogflow and firebase as Google Cloud services. Integration of Face Recognition and the smart door is successful. Smart door will be unlocked if faces are recognized with an average trust of more than 60%, If the face is not recognized, an email notification containing a face image also successfully sent to the house owner. The Google Assistant could also handle user request successfully with a success rate of 92.8% from 147 trials.

Copyright © 2019 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Ivan Surya Hutomo,
Department of Electrical Engineering,
Petra Christian University,
Jalan Siwalankerto 121 – 131, Surabaya, Jawa Timur, Indonesia.
Email: hutomoivan@gmail.com

1. INTRODUCTION

Implementation of Internet of Things for smart home nowadays also concentrates to accessibility and security system. In the past, human used the mechanic door with key and padlock. The disadvantages of this system were the door must be opened with the original key. This system also insecure and nit efficient because the mechanic system of the door could be modified to open the door forcedly, besides human also must bring physical key everywhere so they could access the door.

Furthermore, the development of smart door leads to the addition of electronic authentication features such as NFC and so on. Electronic authentication then developed into biometric authentication. Biometric methods are the use of biological characteristics of living things that are unique and not identical to one another for security purposes. Biometric methods that are widely applied in the smart door are fingerprint and voice recognition.

Integration of Face Recognition with smart door is considered to be more efficient than using NFC and other biometric methods. By using camera-based face recognition, the house owner does not need to make physical contact to open the door because it will open automatically when a face with true authority is detected. The use of face recognition is a modern security technology because this technology will act like a human in

recognizing certain faces so that face recognition technology can be said to be a technology that increasingly approaches human intelligence [1].

Previous research entitled "*Room Access Control System using Face Recognition Image*" by Frans Rotinsulu has not yet integrated face recognition on an actuator and also still uses a laptop and webcam as an interface, therefore the author wants to implement a face detection Haar Cascade and Local Binary Pattern Histogram on an actuator to open and lock smart door.

Previous research also yet not integrated smart door with good and up-to-date user experience and also not integrated smart door with a good notification system. Therefore, in addition to integrating smart door with face recognition, in this study, the author also designed and integrated smart door with a natural user interface where later users can communicate with the door like communicating with other humans. The author will also design a notification system that is good for smart door systems. It is expected that with the use of the Natural Digital Interface and the latest notification system, it will enhance the user experience, especially making it easier for use by ordinary people.

The natural user interface will use the Google Assistant service that can help provide clearer information about the condition of the door at that time including improving the user experience of IoT devices. By integrating smart door lock with face recognition and Google Assistant, users are expected to have a more efficient, safe and interactive way to unlocking the door.

2. RESEARCH METHOD

In the design of Figure 1, the Raspberry Pi 3 will be used as the main microcontroller to control sensors, actuators, including the image processing. One sensor connected to the Raspberry Pi 3 is the Raspberry Pi Camera V2 as a sensor for taking pictures. The image taken will be detected whether the image contains a face or not with the Haar Cascade algorithm. If there are faces detected, the process will continue to the Local Binary Pattern Histogram algorithm to recognize faces in the image. The whole algorithm uses the Python language with the Open Computer Vision (OpenCV) library.

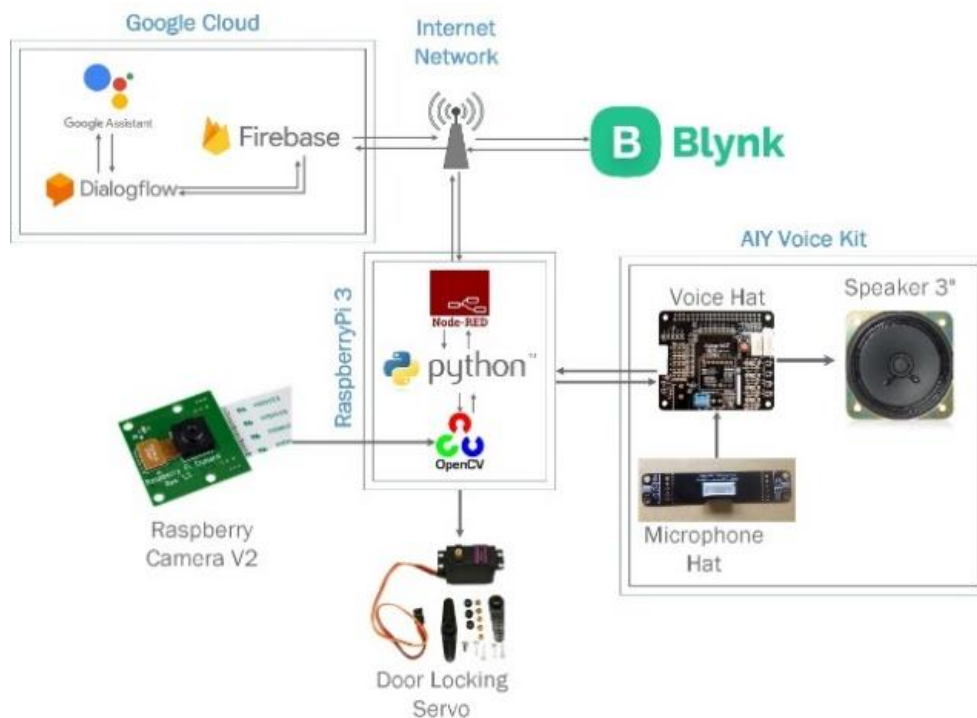


Figure 1. Overall system design

After the image processing step is done, the data obtained from the process is thrown to the Node-Red platform, Node-Red will then send data to the servo motor in accordance with the provisions of the face recognized or not. Node-Red will also send data to Firebase and the Blynk application on smartphones so the house owner could get notifications from the smart door. Dialogflow will then retrieve data from firebase and

forward it to the Google Assistant. By using Dialogflow, users can conduct training phrases that are used to trigger Google Assistant so that the Google Assistant can know the condition of the door at that time.

The Raspberry Pi 3 is also used to control the speaker and microphone with AIY Voice Kit. AIY Voice Kit will work as a Google Home Assistant device at home which is also integrated with the Google Assistant smartphone.

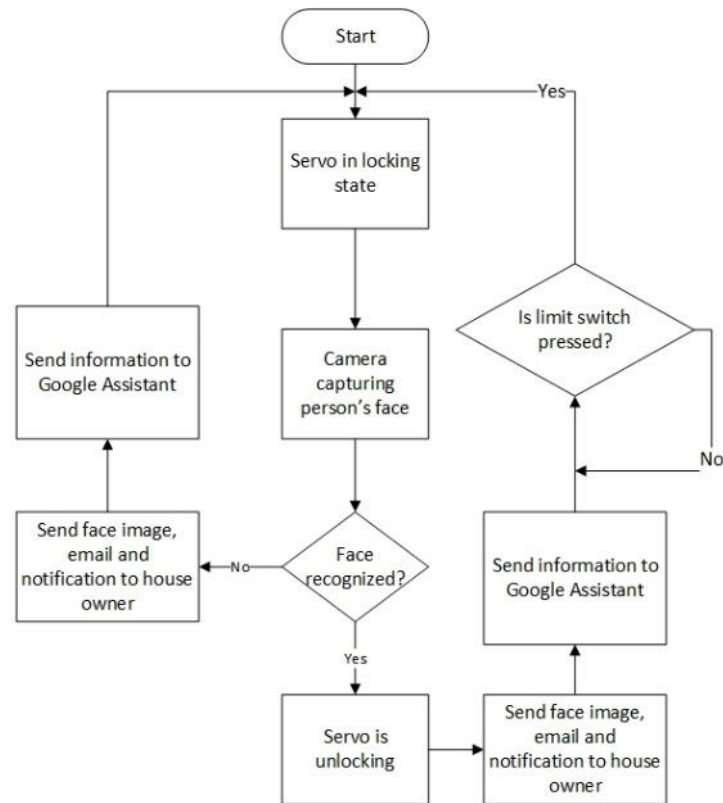


Figure 2. Flowchart of smart door

Based on Figure 2, when the system initiated for the first time, the servo will be in the lock condition, then the camera will capture the face in front of the door at that time and the face will be matched with the database training file. If the face is recognized, the servo will be unlocking the door and then the system will send an email and notification to house owner while sending face ID information and door conditions to Google Assistant. If the face is not recognized, the servo will keep locking the door but the notification will be sent to the house owner. After the servo is unlocked, the system will wait for the limit switch condition, if the limit switch is pressed, the servo will be in the lock position.

2.1. Haar Cascade and Local Binary Pattern Histogram as Face Detection and Face Recognition Algorithm

Haar Cascade Algorithm is used for the face detection process. In general, the Haar-Like feature is used to detect objects in digital images. The term Haar shows a mathematical function (Haar Wavelet) in the form of a box, the principle is the same as in the Fourier function [2]. At first, image processing is only by looking at the RGB value of each pixel, but this method found already ineffective [3]. Viola and Jones then developed it so that Haar-Like features were formed.

Haar-like feature processes images in a grid, wherein one box there are several pixels. Each box is then processed and produces a different value that indicates dark and light areas. These values will later be used as the basis for image processing. The way to calculate the value of this feature is to subtract the pixel value in the white area by pixels in the black area. For moving images such as videos, this process is done discretely by sampling videos at a certain frame rate. Variations in the Haar-like feature [4] are shown in Figure 3 as follows:

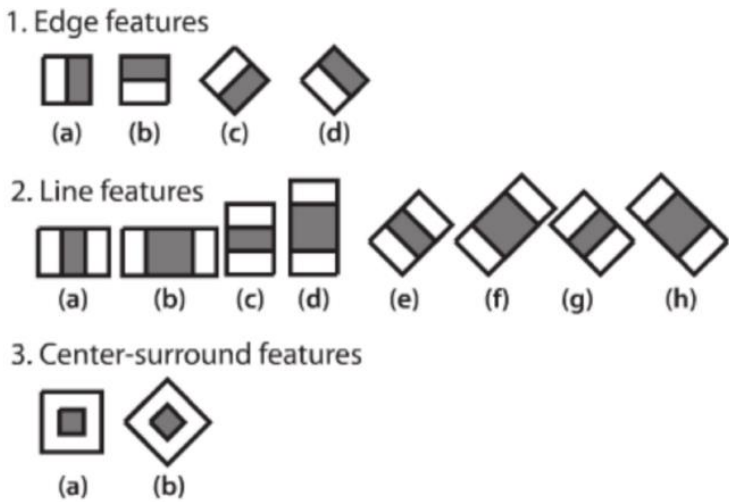


Figure 3. Haar-Like features

If in realtime images that have been converted into grayscale patterns are found as in the picture above, it can be ascertained that the image contains objects.

Face recognition is an advanced process of the face detection process. Face detection can be through photos and videos. By utilizing the training results from Haar Cascade, the results of this process are combined with the image matching process with the Local Binary Pattern Histogram algorithm. With this method, photos that have been learned will be matched with the detection results from streaming cameras. Where in the later, some images in the database are then matched with utilizing histogram values that have been extracted from the image by utilizing the Local Binary Pattern Histogram equation.

The main characteristic of face recognition using this method is the composition of the microtextured-pattern, which is a nonparametric operator that describes the local spatial image. LBPH is defined as a comparison of the binary value of pixels in the center of the image with 8 values of pixels around it. For example, in an image divided by 3x3, the binary value at the center of the image is compared with the surrounding values. By subtracting the pixel value at the center of the image with the pixel value around it and then if the result is more or equal to 0 then it is given a value of 1 and if the result is less than 0 then it is given a value of 0. After that, 8 binary values are arranged clockwise or vice versa and changed 8 bits binary into a decimal value to replace the pixel value in the center of the image. After arranging binarization in a clockwise direction, if one of the binary threshold boxes has a value of 1 then enter the binary value according to its rank, but if 0 then the result is also equal to 0. Finally, add the LBP value.

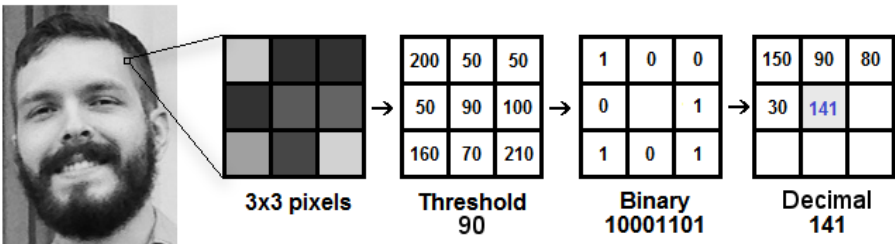


Figure 4. A diagram showing the binarization process of the face's image

To match the owner's face, an equation is used to get the approach of the histogram value which is then used as a predictive value to identify the owner of the face. Therefore, the algorithm output is from the image with the closest histogram. The algorithm must also return the calculated distance, which can be used as a measure of confidence value. Threshold values and confidence can then be used automatically to estimate whether the algorithm has recognized the image correctly. If the confidence value is lower than the threshold, the algorithm has succeeded in recognizing the image.

2.2. Hardware Design

In this research, Raspberry Pi 3 is used as the main microcontroller. In the previous research [5], it was using laptops as the main processors of face images which were considered less portable and less flexible when operated permanently. In that study, it was recommended to replace the laptop platform with a smaller microcontroller that can be applied to various systems, namely Raspberry Pi.

The author uses the Raspberry Pi 3 B because it is considered to be enough to do the face recognition process, besides that in this system the Raspberry Pi 3 is also compatible with AIY Voice Kit provided by Google. Besides that, the Raspberry Pi 3 has an easy GUI with a Raspbian operating system that can be expanded with various platforms such as OpenCV, Node-Red, MQTT, and so on. Raspberry Pi 3 also does not require much power to operate.

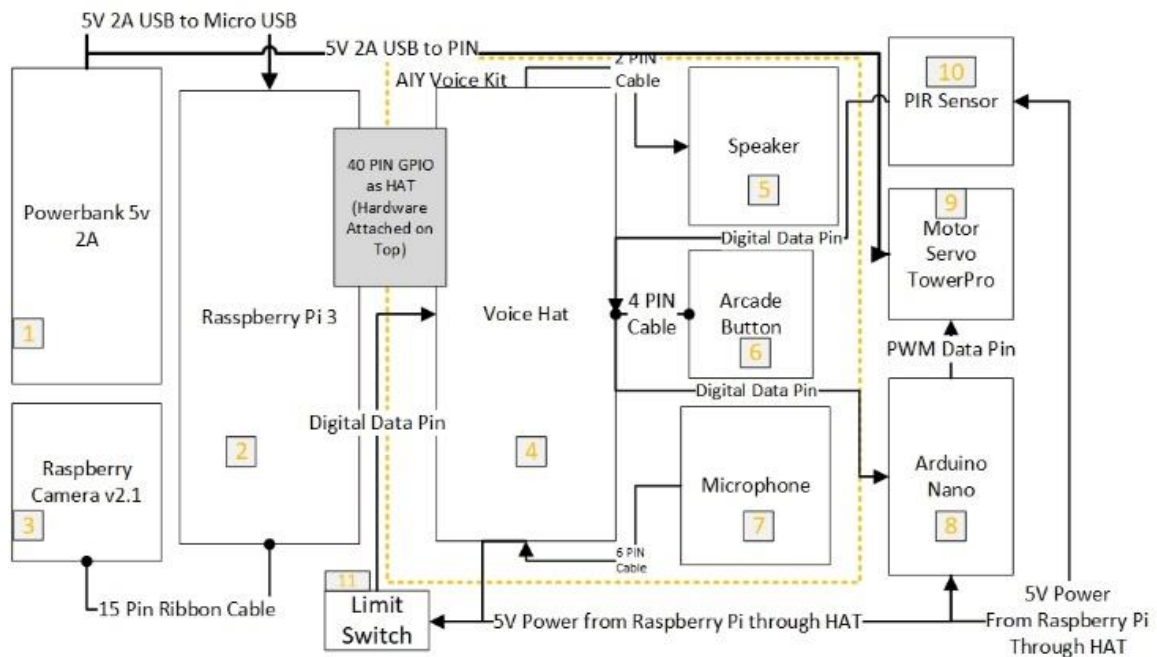


Figure 5. Hardware design diagram

In Figure 5 Raspberry Pi 3 will be powered by power bank 5v 2A (Number 1) using a micro USB interface. The use of this power bank can also be replaced with a 5v 2A adapter that is commonly used for smartphone batteries charging. The power bank will also supply power to the servo motor so that the servo motor (Number 9) can obtain enough torque to turn the deadbolt.

Raspberry Pi Camera v2 (Number 3) is used as the main sensor for face recognition, where this device is connected to the Raspberry Pi 3 using a 15-pin ribbon cable. Voice Hat (Number 4) will be installed on top of Raspberry Pi 3 which is designed as Hardware Attached on Top (HAT). Furthermore, the microphone (Number 7), speakers (Number 5), and the arcade button (Number 6) will be connected to the Raspberry Pi 3 via HAT. While the HAT installed on the Raspberry Pi 3, GPIO (General Purpose Input Output) on the Raspberry Pi 3 cannot be accessed directly, so accessing GPIO can be done through Voice Hat. GPIO Voice Hat is connected to Arduino (Number 8) and PIR Sensor (Number 10).

The use of Arduino (Number 8) in the design of this study aims to overcome the jitter on the servo motor (Number 9). Jitter is an unstable servo movement caused by a bad PWM signal. Previously, the writer connected the servo motor with Raspberry Pi 3 directly, but during simulation, servo motor movement was unstable, especially when the Raspberry Pi 3 performed many multitasking processes. Therefore, in the design of this study, the authors use Arduino Nano as a PWM signal generator for servo motors. Raspberry Pi 3 will only perform digital triggering on Arduino nano and subsequently Arduino which will emit PWM signals to change servo angles.

The use of PIR Sensor (Number 10) in this study serves to detect whether there is a person or not at the front door. If there are people in front of the door, the OpenCV will start image processing, but if there are no people at the door then OpenCV will only prepare the camera. With the PIR Sensor, image processing will

only be done if needed, thus saving Raspberry Pi memory resource usage and prevent temperature raising in Raspberry Pi 3.

The Limit Switch (Number 11) will be installed on the door frame to find out whether the door is closed or not. When the door is closed again, the servo will lock the door, whereas if the door is not closed, the servo will not rotate to lock the door. The following is a description of the smart door miniature that has been assembled (Figure 6):



Figure 6. Smart door assembled miniature

2.3. Software Design

The design of this research will make face recognition of smart door work in general as follows (Figure 7). When the smart door is first initiated, if the user does not press a button to add face data, the system will detect whether there is anyone or not in front of the door. If there are people, the camera will read faces and OpenCV will do image processing to recognize faces. If the face is recognized, the servo will move to unlock. After the unlocking process is done, the data such as door status and face recognition data will be uploaded to the real-time database in Firebase so that the Google Assistant can read through Firebase.

After all the data is stored in Firebase, the system will send an e-mail to the house owner. Furthermore, the person who accesses the door will open the door and the limit switch will detect whether the door is already closed or not. If the door is closed again, the limit switch will be pressed so the servo will rotate to lock the door again. If the face not recognized, the servo will not unlock the door, but attempts to access the door are still recorded in Firebase and a fixed notification email is sent to the house owner that there is an unknown person trying to access the door.

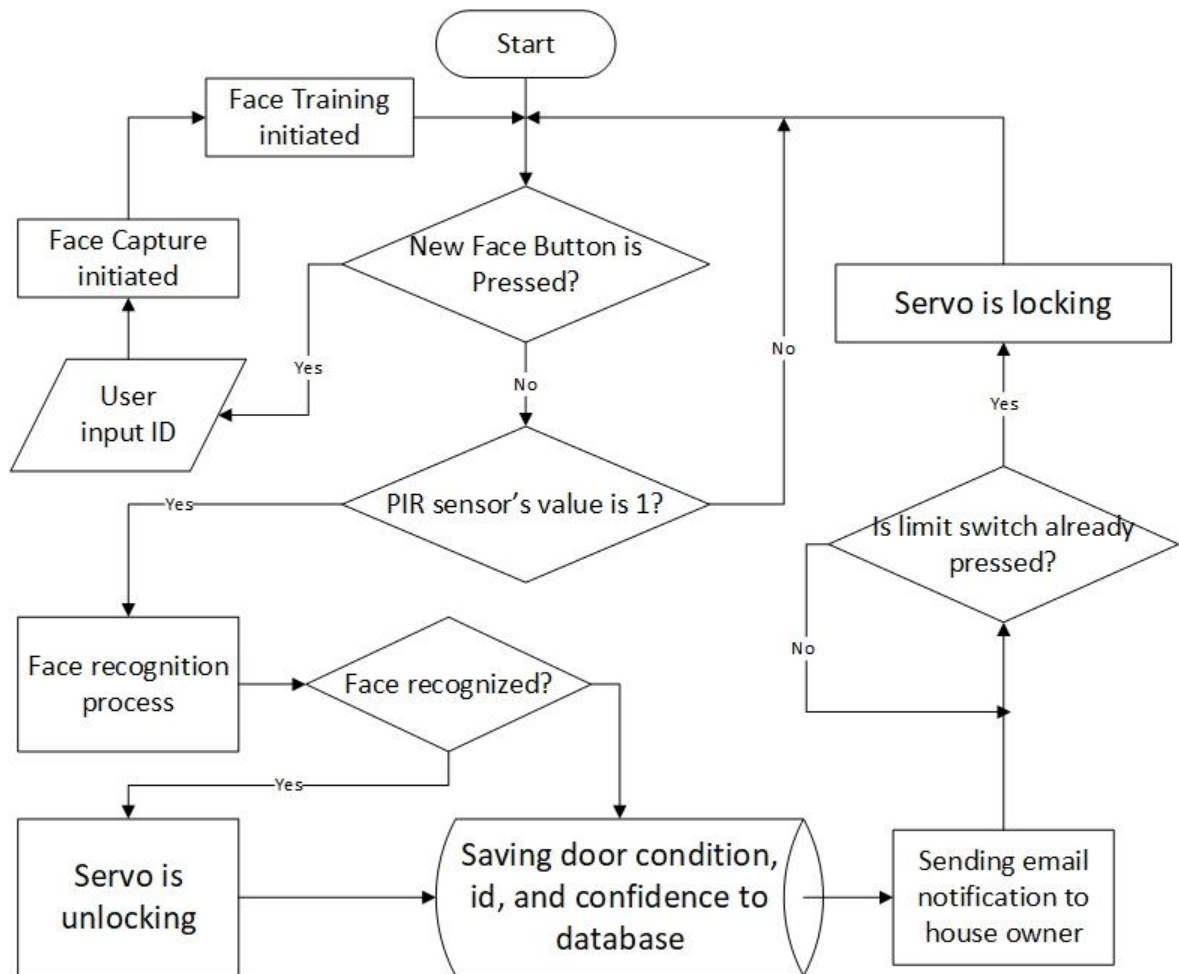


Figure 7. General software flowchart of the smart door

Back to the beginning of the flowchart, if the homeowner wants to add new face data, the homeowner will press the capture button and then enter the number that represents the owner's ID, then with the OpenCV library using face detection, the Raspberry Pi Camera will take 30 sample samples to make face recognition training material.

The training process is actually a process that can be done separately, homeowners can do face recognition training at any time, the training will use a dataset that has been captured before. Because training doesn't need a camera like a capture and recognition process, the process is separate from the main process. Based on functional design in general in (Figure 7). The software design for the smart door lock can be seen in the following block diagram (Figure 8).

In Figure 8, the whole system will be connected to Node-RED which will integrate the whole system. Node-RED itself is a browser-based visual programming language that was developed based on 'flow'. In the design of the software, there are several platforms involved.

Following is an explanation of the schematic of Figure 8, Node-RED is the main regulator and the connector of the whole system. The RED-node will communicate with the Realtime Database (Firebase Server) where Node-RED will provide message information received from Python face recognition programs through MQTT Broker to certain paths in Firebase. Firebase will communicate with Dialogflow which will be integrated directly with the Google Assistant service on the user's smartphone. This communication uses a full fill request where we can create a function that causes Dialogflow to retrieve data on certain paths in Firebase.

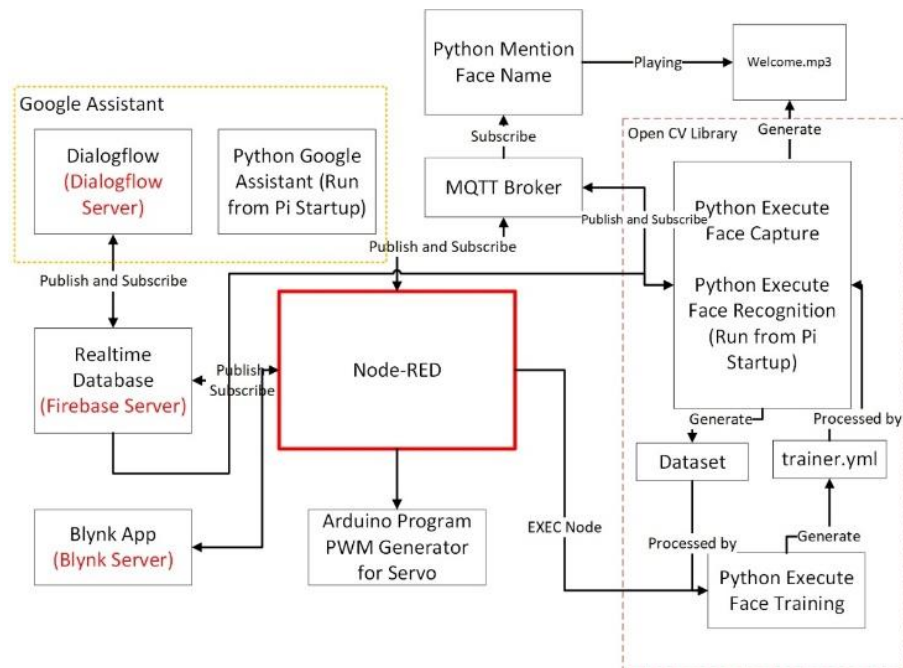


Figure 8. Software relationship block diagram of smart door

Node-RED will also communicate with the Blynk application, Node-RED could communicate with Blynk using virtual pin from Blynk. If Blynk orders to lock the door, Blynk will give a message on the topic of a particular MQTT so that the Node-RED will control the servo based on the message from MQTT. Face recognition programs are coded using Python and exchanging data with Node-RED via messages on MQTT brokers. Python Face Capture and Face Recognition are a unified program because they both require the same face detection library (Haars Cascade). If face capture is done then Python will generate 30 Image datasets for one face. This dataset is then carried out by a training process using the Local Binary Pattern Histogram to make a face recognition pattern histogram where each face will have a unique pattern histogram. The results of this training will be called `trainer.yml`. This file will be used in facial recognition programs to recognize faces. If a face is recognized, for example, the face id is 1, then Python will communicate directly with Firebase to retrieve the name at index 1 in a particular path in Firebase. The name will later be generated by Google TTS library with the `welcome.mp3` file.

3. RESULTS AND ANALYSIS

In this chapter, the simulation will be conducted on the process of taking the dataset, first the distance of the camera to the resolution of the dataset, then the effect of the resolution produced by the length of training time and accuracy of facial recognition. Next will be simulation the face recognition parameters in the Haar cascade namely `ScaleFactor` and `MinNeighbor`. After getting the best `ScaleFactor` and `MinNeighbor` values, then simulation of training duration, number of dataset collection, and also the accuracy and speed of the system in recognizing faces. Simulation is also carried out to test the integration of Google Assistant with smart doors using Dialogflow. Here are the tests conducted in this research:

3.1. Simulation of Camera Distance to Resolution, Training Time, Accuracy, and Face Recognition Time

Based on the simulation, the distance of the camera has an effect on the resolution of the dataset produced. The resulting dataset resolution also has an effect on training time, accuracy, and face recognition time.

Table 1. Comparison of camera distance, resolution, training time, accuracy, and face recognition time

Face Distance to Camera	Dataset Resolution	Training Time	Accuracy	Face Recognition Time
45 cm	191x191 px	3.65 s	42.20%	3.031 s

30 cm	320x320 px	9.45 s	55.30%	2.49 s
-------	------------	--------	--------	--------

Based on the table above the closer the face distance to the camera the higher the resolution of the dataset produced. The higher the resolution of the dataset, the higher the training time required, but the accuracy of facial recognition increases to 55.03% and the face recognition time is faster to 2.49 s when compared to smaller resolutions. The high-resolution causes LBPH to be able to map histograms with more detail on the face of the dataset so that it can ultimately improve the accuracy of face recognition. From the above tests, the face recognition distance is 30 cm.

3.2. Simulation the MinNeighbor and ScaleFactor Parameters for Face Recognition Accuracy and Time

Simulation is done to find the most optimal *ScaleFactor* and *MinNeighbor* parameters based on the best accuracy and the fastest face recognition time. Average accuracy and time were taken from 10x face recognition experiments. In the first test, the *MinNeighbor* value is fixed with a value of 3 and the *ScaleFactor* value is changed. Then after the most optimal *ScaleFactor* value is found, the *ScaleFactor* value is fixed and the *MinNeighbor* value is changed. Based on the most optimal parameter simulation for *ScaleFactor* is 1.5 (Table 2) and for the *MinNeighbor* is 2 (Table 3).

At *ScaleFactor* 1.1, the average time has the worst number of 6.1227 seconds (Table 2), this can be analyzed because the reduction process to match the dataset is not large enough so that the reduction occurs repeatedly and requires a longer time to match the dataset. On increasing each value, the time produced is faster and at 1.5 has the best value. In the *ScaleFactor*, the face recognition *ScaleFactor* process has increased time because the resulting reduction value is too large so it is unable to match the dataset.

Table 2. Comparison of ScaleFactor values with Face Recognition Accuracy and Time

Scale Factor	MinNeighbor	Accuracy Average (%)	Time Average (s)
1	3	Error	
1.1	3	46.93	6.1227
1.2	3	41.76	3.973
1.3	3	50.97	3.111
1.4	3	44.08	2.423
1.5	3	58.39	2.059
1.6	3	31.537	2.204
1.7	3	55.52	2.38
1.8	3	26.29	5.458
1.9	3	Error	

Based on simulation (Table 3) the best *MinNeighbor* value is 2, where the average face recognition accuracy can reach 62.04% with an average time of 2,093 seconds. *MinNeighbor* in this research experiment does not really affect the accuracy and speed of face recognition. This can be analyzed because the face in this experiment is close to the camera that is 30 cm and the face has filled the entire camera frame so that the possibility of a false positive is very small and ultimately does not affect accuracy. However, the smaller *MinNeighbor* value, the time to recognize faces is faster even though it does not change significantly (Table 3), this is because when *MinNeighbor* is smaller, the system will be more sensitive in recognizing faces (increasing the possibility of false positives) so that the time to recognize faces is faster.

Table 3. Comparison of MinNeighbor values with Face Recognition Accuracy and Time

Scale Factor	MinNeighbor	Accuracy Average (%)	Time Accuracy (s)
1.5	1	42.032	2.0484
1.5	2	62.04	2.093
1.5	3	59.06	2.081
1.5	4	51.71	2.118
1.5	5	47.827	2.0874
1.5	6	57.12	2.105
1.5	7	58.11	2.024

1.5	8	50.71	2.079
1.5	9	46.18	2.171
1.5	10	58.47	6.037

3.3. Simulation to Find Average Confidence as a Threshold for Unlocking Smart Doors

This test aims to determine the ability of the system to differentiate faces that are recognized and faces that are not recognized. This test needs to be done because it affects the authorization to open the door. If a face that is not registered in the dataset is recognized as the face of another person which registered in the dataset, the door lock can be opened accidentally. In this test, the dataset only consists of 3 registered respondents, while another respondent will not be registered in the dataset and the system should not be given authority to open the door lock (not recognized).

Table 3. Simulation of the unknown dataset to find unlocking threshold

Dataset Registered Face	Names	Accuracy Average (%)	Time Accuracy (s)
Ivan, Michael, Gavrielle	Ivan	65.34	3.876
	Michael	73.63	3.972
	Gavriel	67.48	4.047
	Jischak	44.18	5.252
Ivan, Michael	Ivan	63.39	3.474
	Michael	70.711	3.477
	Gavriel	55.34	3.496
	Jischak	50.61	3.388
Ivan	Ivan	62.59	2.642
	Michael	50.78	3.171
	Gavriel	53.41	3.019
	Jischak	51.57	2.983
False Confidence Average		50.98166667	
True Confidence Average		67.19016667	

From the results of simulation in the table above, it was found that the results on faces that are not in the dataset are never unknown but are always recognized as faces in the dataset. In the first test, *Jischak* was not captured in the dataset, the system should not have recognized the face, but *Jischak's* face was recognized as *Michael*. Even so, the average confidence generated was not as good as *Michael's* original face, which was 44.18% (*Jischak*) versus 73.63% (*Michael*). In the second test, the faces of *Gavriel* and *Jischak* were not captured in the dataset but both were identified as *Michael* with false confidence - 55.34% and 50.61% respectively while *Michael's* original face had true confidence of 70.711%. In the third test, only *Ivan's* face was captured in the dataset, however, the faces of *Michael*, *Gavriel*, and *Jischak* were identified as *Ivan's* face with false confidence - 50.78%, 53.41%, and 51.57% while *Ivan's* face had true confidence of 62.59%.

Through the three tests, each true confidence and false confidence were calculated on average and produced average false confidence of 50,981% and true confidence of 67,176%. From these average results, to prevent an unknown face from opening a door lock, the average confidence to open a door must be above 50,981%. However, for the face of the authorities to be able to access the door, the average confidence must be below 67,176%. By paying attention to the highest average false confidence (55.34%) and the lowest average true confidence is 62.59% then the average confidence to open the door can be determined 60%. Using this value, a face with confidence greater than 60% is considered recognized and can open the door, while faces with confidence less than 60% even though recognized as a face in the dataset (false confidence) do not have the authority to open the door lock and are considered unknown in the system.

The following is a picture of known and unknown datasets where faces that should be unknown are recognized as faces of people in datasets with lower average confidence. Figure 9 is a picture that shows the original face that was identified as Ivan with an average confidence of up to 70% (Figure 9 left). While in figure 9 right, the face of the unknown dataset is recognized as the face of Ivan (known dataset) with a lower average confidence of only 33%.

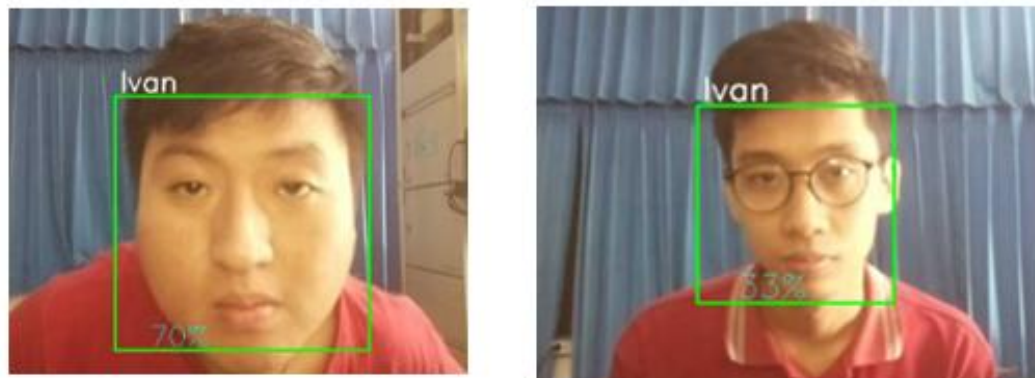


Figure 9. Differences in average confidence in known datasets (Left) and unknown datasets (Right)

3.4. Google Assistant Integration Simulation

Simulation is also carried out to test the success rate of Google Assistant in responding to orders from users. The success rate can be seen in the session flow from Dialogflow (Figure 10):

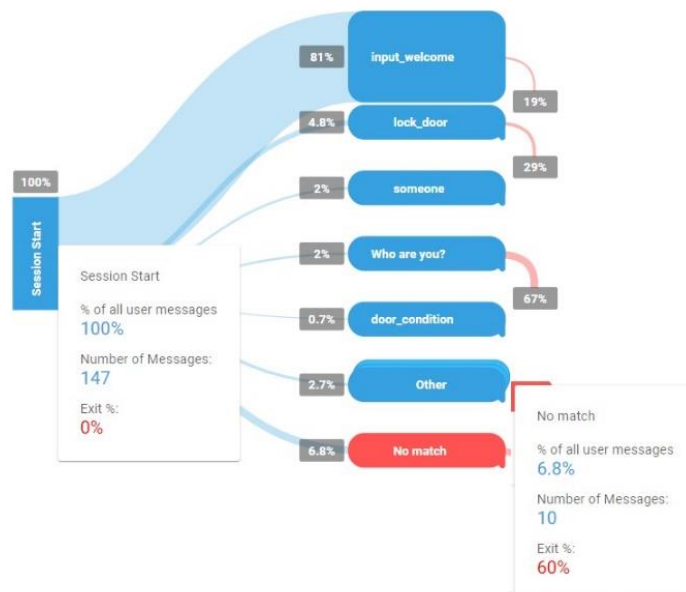


Figure 10. Session Flow that shown the success rate of Google Assistant

Based on the session flow (Figure 10), from 147 experiments conducted, the success rate reached 93.2%. Google Assistant's failure to recognize user speech is only 6.8% (10 trials out of 147). This whole test shows that Dialogflow's integration with Firebase, Node-RED and the author's Google Assistant account have been successful.

4. CONCLUSION

From the entire assembly, manufacture, and simulation of smart doors integrated with face recognition and Google Assistant, the following conclusions are obtained:

1. Face recognition using Haar Cascade face detection and LBPH face recognition was successfully integrated on the smart door with the most effective dataset taking distance of 30 cm, MinNeighbor 2, and ScaleFactor 1.5. The number of effective faces stored in the database is 3 faces, more than 3 faces the system starts experiencing false recognition. If the average confidence of the face is more than 60% the smart door will unlock while if it is less than 60% it will be classified as unknown.

2. Integration of Google Assistant with smart door works well. Google Assistant can be used to retrieve information, control smart door devices by utilizing Dialogflow and the realtime database from Firebase. Based on the simulation, it can be concluded that Dialogflow and Firebase are reliable enough to be utilized and integrated with smart doors with a success rate of response reaching 93.2%


ACKNOWLEDGEMENTS

We express our thanks to Electrical Engineering Petra Christian University which providing us facilities to learn and increase our knowledge. Thank you to my beloved supervisor professor **Handy Wicaksono** which always give motivation, knowledge, references and tremendous moral support.

REFERENCES (10 PT)

- [1] M. P. Beham and S. M. M. Roomi, "a Review of Face Recognition Methods," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 27, no. 04, p. 1356005, 2013.
- [2] P. Purwanto, B. Dirgantoro, and A. N. Jati, "Implementasi Face Identification Dan Face Recognition Pada Kamera Pengawas Sebagai Pendeteksi Bahaya," *eProceedings Eng.*, vol. 2, no. 1, Apr. 2015.
- [3] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, p. I-511-I-518.
- [4] S.-K. Pavani, D. Delgado, and A. F. Frangi, "Haar-like features with optimally weighted rectangles for rapid object detection," *Pattern Recognit.*, vol. 43, no. 1, pp. 160–172, Jan. 2010.
- [5] F. Rotinsulu, "Sistem Kendali Akses Ruangan menggunakan Citra Pengenalan Wajah," Petra Christian University, 2015.

BIOGRAPHIES OF AUTHORS (10 PT)

	<p>Ivan Surya H was born in Surabaya, Indonesia in 1997. Received his bachelor degree in electrical engineering at Petra Christian University, Indonesia in 2019. He then continues his study of master degree in Electrical Engineering and Computer Science at National Chiao Tung University, Taiwan. His field of interest includes Artificial Intelligence, robotics, and automation.</p>
<p>Second author's photo(3x4cm)</p>	<p>Xxxx (9 pt)</p>