G.LOCK

Author: Chinedu Ojukwu Joel Osei Omar Alhait
Electrical and Computer Engineering, Carnegie Mellon University

Abstract

A system capable of providing extra security for residents of a household. This system will be using a neural network in order to detect when anyone is at the front door and identify those who actually live in the house. Using a camera, the individual's face becomes the key to the house instead of a physical key. There will also be a mobile application which will be used to update residents of the house with a log of everything happening, and will give them the power to both control what is happening with the lock.

***Index Terms*—**Camera, Electric Strike, Facial Recognition, Hardware, Machine Learning, Networking, OpenCV, Security, Server, Streaming, Web App

I. Introduction

The three of us building this project are roommates who live in a 5 bedroom house with 7 total occupants. Within this household there are always people entering and exiting the front door, which causes for a security issue. It isn't very effective for us to each have a physical key and unlock and lock the main door whenever we plan on leaving or entering the house. So we have devised a system that will allow us all to enter and exit the house immediately, while still being secure at the same time. We could have purchased a nest or ring system for our front door. However, these would only solve half of our problem, and would be a bit more expensive as well. None of these systems are able to use facial recognition to quickly identify residents of the house. They are mostly used to keep track of when people are at the door, and give people control to unlock/lock whenever they want

The G.Lock will solve all these problems and more. One of our biggest goals is to be efficient, fast, and convenient. There is no point of having an unlocking system that would be significantly slower than just using physical keys on a normal lock. We estimate that it takes an individual usually about 10-15 seconds to walk up to the door, find their key, and finally unlock the door and enter the house. We want to be better, so the G.Lock will utilize the most efficient machine learning algorithms to detect a face within less than a second. This allows for the machine learning processing to take less than 10 seconds to identify an individual. As a result, we expect G.Lock to take at most 10 seconds to recognize if an individual is an inhabitant of the house or not.

II. Design Requirements

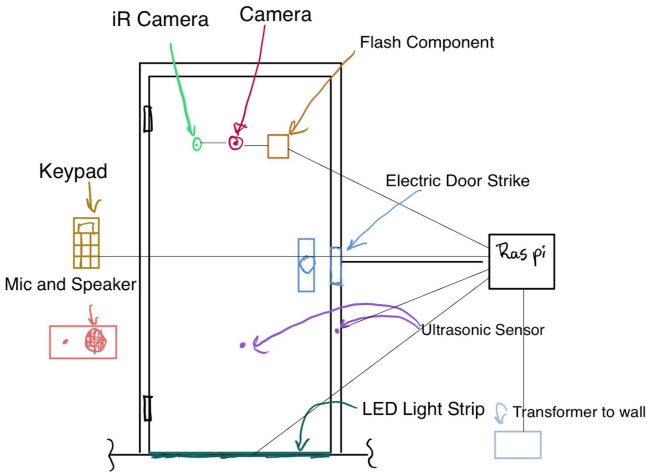
Our system will have the following requirements:

- Face detection success - Day: 90%; Night: 85%;
  - Should be able to detect a face 90% of the time during the daytime, and 85% of the time in the nighttime. This is due to the low light which may affect the images produced by the camera.
- Efficient System power
  - We aim to power every component of our circuit using both batteries and a wall outlet. Our goal is to have the system be able to last 48 hours on its own, using no more than 20 Volts.
- Neural Network Accuracy - Day: 85%; Night 75%;
  - Our overall system should be able to recognize residents of the house 85% of the time in perfect lighting situations, such as daytime and indoors. However, only 75% in bad lighting situations.
- NO FALSE POSITIVES
  - For a security system that allows access to the front door it is imperative that we do not have false positives. Meaning that it is ok for us to deny access for people who actually live in the house, but never ok to allow people who don't live in the house access.
- FailSafe Unlocks
  - Residents will be able to try again if for some reason access to the house is denied by the neural network. If for some reason it denies them again we want them to have multiple methods of still entering the house. They can either use the mobile application, or use the keypad. Although, there will be a keyhole, G.Lock aims to eliminate usage of the key entirely.
- Mobile application unlock - 10 seconds;
  - Users with the mobile application interface should be able to unlock the door from their phone as quick as possible. We are setting this requirement to 10 seconds as there are many different factors such as latency and connection issues.

In order to fulfill these requirements we will be building our system with two things in mind: accuracy and efficiency. It is our intent to keep the residents of the household in mind and have their convenience in mind. There is no point of building a system that would be slower than the process of unlocking the front door with a key. We plan to run multiple tests in order to accomplish these goals. This means breaking down the system into their subcomponents, making sure each does what it needs to do. Finally, combining everything and running through the entire process many many times.

III. Architecture and/or Principle of Operation

Our system is composed of three main components: the camera and facial recognition software, the mobile application, and the physical locking circuit. We will break these three down with great detail in this section and section V.

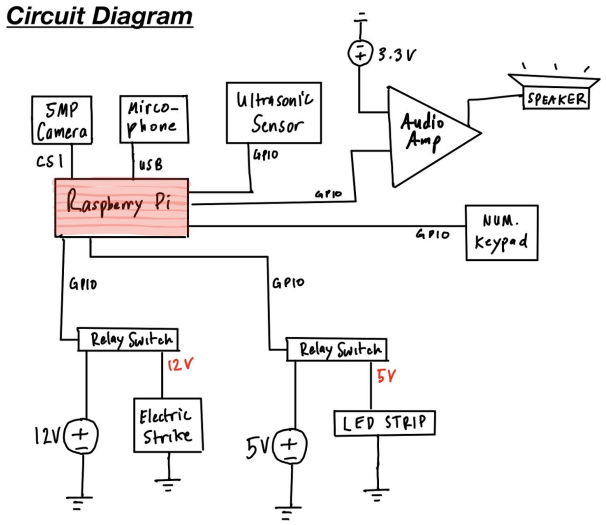


(a)

Fig. 1. (a) Placement of every component with respect to front door.

In the diagram shown here, the system will be powered on by the ultrasonic sensor, which will send a turn-on signal to all the necessary analysis components. This will activate the facial recognition through the cameras and flash, and the LED Light Strip will shine accordingly to the verification. If the face is verified correctly, the electric strike slides and props the door open, and the LED's will turn green and motion the user inside.

In non-ideal cases, or if the inhabitant isn't in front of the door, the microphone system could be used to have a conversation with a house inhabitant through the G-LOCK, or the keypad could be used to gain manual access into the house. This could be used for people who access the house frequently but aren't inhabitants, like housekeepers or gardeners. The electric door strike won't be the only other source of entry, as someone can still open the door by opening the small lock inside the doorknob itself.



(b)

Fig. 2. (b) Zoom in of the specific circuit components

In our circuit diagram, our Raspberry Pi will serve as the center of communication for all our parts. They are all fed into the Pi through respective GPIO pins, with the exception of the camera, which is fed through the CSI ribbon port. The speaker will need its own voltage source to power it, as well as the electric strike and LED strip.

Essentially, we want the Raspberry Pi to serve as the brain of our system, as it will receive and deal with input from all the GPIO pins, as well as run the expensive facial recognition algorithm. Since there are worries for the performance of such a small computer for long intervals of time, we are ready to deploy our system on two Raspberry Pi's instead, with one focused on the computer vision and one dealing with all the GPIO devices.

IV. Design Trade Studies

A. Tradeoffs

We had many different options that we came up with when brainstorming about which lock we would use for the door. We had 4 options to begin with: Magnetic Lock, Electric Strike, solenoid and door latch, and deadbolt lock with a 3d printed cover. We needed the most effective, most efficient, and most secure lock for our system. This immediately took out two options, the deadbolt and door latch.

For the deadbolt we would have to 3d print something that would go over a door that already had a deadbolt lock on it. This was a good idea because it would allow our system to mobile, where instead of building everything from scratch including the door, we could just manipulate what was already there. However, we would have to control it with servos in order to unlock/lock the door and this deemed to be extremely inefficient and not that secure. Both servos would have to be powered, and if something went wrong with one of them, the entire system might fail. This was a similar reasoning we used to rule out the solenoid and door latch. That combination also seemed to be not very secure, as the only thing keeping the door locked was a door latch.

The final decision relied on either a magnetic lock, or an electric door strike. Both of these seemed like viable options and to be completely honest either would have worked to build this system. However keeping our main requirements in mind, we were able to find an electric door strike that was both cheaper, smaller, and required less power than most magnetic locks. So that is exactly why we are using an electric door strike to keep the door locked.

Another huge design decision we had to make was which camera we wanted to use. The camera is probably one of the most important parts of this entire system. If our camera is bad or low quality, the machine learning would never be able to identify residents of the house. So we took our time analyzing which camera to get. Our three options were between a raspberry pi camera, the Pixy camera, and finally an iR camera. Each came with its benefits and drawbacks.

The pixy camera came with its own built in object tracking as well as a pan/tilt system. So it would have been able to tilt along the x and y axis, which is something that looked extremely appealing to us. However the object detection did not work with faces, and the camera was a bit more expensive than every other camera. For the iR camera, it was the lowest quality of all the cameras and only worked by picking up infrared signals. This would be a great addition for nighttime usage of our system, however we want to train our neural network with images that are as close as possible to being the same. So we finally decided to use the high quality 8 megapixel raspberry pi camera. Although it came with no pan/tilt, it was built for the pi which gave us more flexibility in what we could do with it.

One of the most important design choices we made was what machine learning model to use. For facial recognition there are a plethora of different options that could have used. We were always thinking of neural networks as one of our main options, but we looked at many different things. These included eigenfaces, eigenvectors, and pure statistics. Eigenfaces and eigenvectors relied on principal component analysis which is basically dimensionality reduction. They work great for a facial recognition that would be used to find so many different faces. However, since we are only working with 7 individuals, we believed fine tuning a neural network would work far better for us in this specific situation.

V. System Description

The G.Lock system consists of three main components. The camera and machine learning, the physical circuit, and the mobile application interface. These three components will have their own separate tasks that they must accomplish, but will be able to communicate with other parts of the system through our main hub, the raspberry Pi. We will be breaking down each of these components and describing how they will work together to get G.Lock working.

A. Camera & Machine Learning

The bulk of our work will be put into creating a machine learning model in order to properly identify residents of the house. This means that this specific subsystem will be responsible for first recognizing when someone is at the door and then, capturing a representation of this person and using that representation to determine if they are indeed a resident of the household. In order to accomplish this goal there are three subcomponents that we must describe. The camera and the purpose it serves, how we will be doing facial detection, and finally the machine learning behind everything.

1) Camera

The camera we will be using will be a 8 megapixel camera capable of taking photographs of 3280 x 2464 pixels and video up to 1080p 30fps, 720p 60fps and 640x480p 90fps resolutions. We will mainly be utilizing the camera at its highest resolution in order to obtain the best picture quality for our facial detection and machine learning. The camera on its own will not be doing much but is an extremely important part of this project and we wanted to make it a point of emphasis. In order for our entire system to work properly there are many factors that need to be considered with the camera. These include, the quality, its orientation, position, angle, and even more.

We plan on having the camera positioned on the outside of the door and angled down by about 15 degrees in order to have a very wide view of everything. It will be on the upper third of our door because the average height of a human is

about 5 feet 4 inches. In our household the average height is almost 5 feet 9 inches. The average front door is about 8 feet tall. So in order to get the best images and have more accurate data for our machine learning, we want images where the camera would ideally be positioned perfectly in front of the individual's face. We originally planned to have a sliding rail in order to make sure that with every person who walked to the door the camera could move up and down to be perfectly positioned. This is because we really want to focus our system on being diverse. We want to design our system to not be biased towards a certain group of people with specific heights, or skin tones, or ethnicities. We want to build a system that will work with everyone. However, we found that this would be a bit of extra work and ultimately unnecessary. We don't need the camera to be perfectly positioned as we can crop out the background and focus entirely on the image within our code. So this led us to come up with another idea, where instead of allowing the camera to slide up and down, what if we gave it the ability to tilt up and down. Although this might help a bit, we deemed the work necessary to create this tilt with a few servos, would be more than the improvement of accuracy it would give us. As a result for the camera the final system we came up with is a camera on the outside of the door positioned at about 6 feet up, so people will look up to it or straight at it. We will try to have people stand about 3 feet away from the door and look directly at the camera.

2) Facial Detection and Tracking

The next part of this component that is also extremely important is facial detection and tracking. This basically means that we will be using the camera to do two very important things. First, detect if there is a human face in its frame, and then, following that face by taking about 2 frames per second of the face to make sure that it is tracked until the verification is complete. Most of the work for this part of the subsystem will be handled by an open source framework that we are using called OpenCV. This framework will use Haar feature based Cascade classifier to determine if there is a presence of a face in the image provided by the camera. This works by first converting the image into an array of 300x300 pixels, which is simply a tuple of three numbers from 0 to 255. This array is then analyzed by looking at groups of pixels in order to find cascading features, such as a face or eyes.

We will be doing most of this in python and using OpenCV. With this we have successfully detected the presence of a face in front of our door. The next step is to continually track the face until we can verify they are a resident. This array of pixels is cropped down to be focused only on their face and then fed into our Neural network for processing. We want to keep track of that individuals face for multiple reasons. First, it allows us to know if we should continue the verification process. If there is someone who just came to drop off a package and left, we don't necessarily want to perform the whole process on them if it is unnecessary. The second reason, would be if our initial image isn't the best image to identify the person, we plan on taking about 3 images per second and will just test multiple images before concluding whether or not the door can be unlocked. All these images will be saved locally to the Raspberry pi and will be accessible by the neural network. If the person at the door is indeed a resident of the house we plan on possibly updating the library of initial training data for that person in order to potentially improve the accuracy.

3) Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. We will be using machine learning to identify faces once they have been detected. We will have a neural network model to accomplish this goal. Our neural network will take in a 300x300 array and output a feature vector of an 128 embedding as described by google [5]. We plan on using a neural network with 2 hidden layers. Each layer will be responsible for analysing the 90,000 pixels and figuring out which things are important, and what is not. However, we will definitely be testing many different models and it is hard for us to say right now with great certainty what exactly our neural network will look like. Our prediction is that it will take this array of 90,000 pixels as input, and output a final vector. There are so many variables that need fine tuning in order for this system to be as accurate and efficient as possible. It might require us to have lower resolutions in order to speed up our computation. We will also have initial weights and biases based on what we determine to be important. We also are starting initially by having the 128 embedding be our output goal, however we understand that since there are only 7 individuals being classified, it is possible that we might not need all 128. It will be up to our neural network to decide what is most important when being trained. It will up to us to also test what minor changes result in significant improvements in speed and accuracy. We might have to use 100x100 pixel arrays as inputs instead, or add another hidden layer. There are alot of variables that need to be fine-tuned. The classification process for us will be relatively simple. Once we have the feature vector, all we need to do is check our database of feature vectors and check if this one exists to some extent within our database. This part has to be taken with care as it is possible for false positives to come up if we have a relatively large delta for classification.

B. Lock Circuit

In order to automatically unlock and lock the door when someone is correctly identified with the machine learning, there must be some type of circuit connected to the lock which would convert that digital signal into an activation voltage for a lock and unlock the door. We have outlined a possible

circuit to handle this. There are 2 main parts: The raspberry Pi, which will serve as our main hub to control everything, and finally the other circuit components that together will be connected in order to build this system.

1) Raspberry Pi

RasPi will serve as central processing hub and will be implemented in Python since the language has support for hardware control, network interfacing, and can be used with OpenCv. The language itself will is easy to use. The Pi will handle network traffic between RasPi server and WebApp clients using the Flask Python Framework. We chose Flask partly because of previous experiences with the framework but also because of it is easy to use and extend. The idea behind Flask is to build a solid foundation for web applications of different complexity [9].The Pi will control hardware through GPIO Pins and USB Ports. They provide easy control of the systems components and will allow the Pi to interact with the outside world.

The ML Test Suite which includes training with labeled images and testing with new images will also be hosted on the Pi. This will handle polling each image from the camera and sending it to the MLK test suite in real time.

Two Raspberry Pis might be used instead of one for our overall system depending on the performance we measure this is further explained in the risk mitigation section.

2) Circuit Components

Since some component sin our system require outside voltage sources to be power itself, our RasPi cannot provide enough power to all these components (Electric Strike, LED Strip, Sound Amplifier). See Figure two for wiring details. We will provide the 12V, 5V and 3.3V needed respectively with outside power banks so they can function for extended periods of time. Since the strike and LED strip need a voltage to turn on, we added relay switches that are connected to the power sources and are connected to GPIO pins that serve as switches to turn on these components. The audio amplifiers are connected to a 4Ohm speaker which will play audio from the RasPi. The rest of the circuit components are connected and powered by GPIO and USB pins that provide sufficient power to each component.

C. Mobile Application

We will have a user interface for our system that will be on mobile devices. To write this application, we will be using React Native, a language created by facebook that allows for mobile development on android and Apple at the same time. Although our entire house has iphones, we decided we want to be diverse with our system. This means that we are diverse in every possible way, our system should be able to interact with people who are of different heights, skin tones, ethnicities, and especially people who also own different kinds of phones. There are a lot of different features that we have planned to implement within this application. We have grouped them into two main focuses: Control and communication.

1) Controlling Lock

The system we are creating is supposed to give more control to the inhabitants of the house. We want to allow the residents of the house to be in complete control of their front door and everything that happens with it. This means that they can unlock the door right from their phone even if they are halfway around the globe. To accomplish such a feat we plan on using the server that is hosted by our raspberry pi 2 to listen for clients which will be hosted on the application. Once a client is connected to the server it should be able to send an HTTP request to the server. This is done through POST calls, and a special React Native call, React Requests. Once received the pi will verify where the request is coming from, and if valid will send a signal to the circuit to unlock the door. This will also work as another fail-safe in case there is a problem somewhere else in our circuit. We are building this system with the understanding that things don't always go perfectly. For example, if there is some strange blizzard happening near an individual's house and the camera is unable to identify them, they could try the keypad. However, if the keypad also doesn't work or they forgot the passcode, residents will be able to just use their phone to unlock the door. Although the door will not be unlocked immediately due to unforeseen internet problems and things of that nature that could happen, based on our requirements we came up with a reasonable time in which the mobile app should be able to unlock the door: 5 seconds.

Another aspect of having complete control of the front door is understanding everything that happens even when one is not there. So there will be a log that will be kept of every event that happens with the front door. An event is defined as:

- The system is activated by the ultrasonic sensor
- A face is detected in the camera
- Every final decision made by the neural network
- Remote Locks/Unlocks by other users with the app

So whenever either of these things happen they are stored locally on the Pi as a simple .txt file with whatever extra data might be attached to them. For example, final decisions made by the neural network would also include who was identified and the door was unlocked for. If the final decision was negative we would save the image of the individual who was at the door (This is one of our stretch goals). These will be uploaded to the server and clients will be able to pull that information in order to have a log of all events. This gives complete control to the residents of the house. One of our biggest stretch goals is the ability for people to add an individual to the database in order for them to also be able to enter the house. This will require collecting training data through the app, which is simply a bunch of images of that individual's face. This is then used to retrain our neural

network and added to our current database for classification. This will take a lot of extra work so will be a very far fetched stretch goal of ours.

2) Communication with system

The mobile application will also allow for residents to communicate with individuals at the door through the application. This means that they will be able to do three things: See what is happening as a livestream, listen to messages from individuals at the door, and also send voice messages to the people at the door. In order to accomplish this we will be using and external API that will use the raspberry pi to create a server specifically for streaming audio and video back and forth over the web. This will simulate real time conversation between whoever is on the application and whoever is at the door.

VI. Project Management

A. Schedule

The schedule is provided in Figure 2 of the Appendix. We plan to have a working prototype that consists of a very basic breadboard implementation of the system one week after spring break. From there we will work to fully optimize the system and connect the components that will be in the main system. We will then take two weeks before the demo to integrate all of the subsystems together and place them directly onto the door and the frame. We also plan to run benchmarks to quantify the system performance and analyze how requirements are met.

B. Team Member Responsibilities

Chinedu Ojukwu - Primary responsibilities include main Raspi software logic and hardware control. This includes implementing the functionality that will control data sent to and from the pins to each circuit component (strike, sensors, output devices). Also will be in charge of circuit creation and connecting all the needed circuit components to the RasPi. The main software module will be in charge of polling frames and sending them to the ML test suite. Upon verification, the software will actuate the necessary components and package data to be sent to webapp.

Joel Osei - Primary responsibilities include creation of the Web App in react native and implementation of networking between the RasPi and client. This includes sending data using HTTP requests from the RasPi server over a local area network to web clients. Will be in charge of implementing the stretch goal of streaming live video to the web client from the WebApp. Also in charge of implementing web app features which include user management (adding labels and associated images to training model), system log viewer, remote lock control and notification viewer . Also will assign in ML algorithm.

Omar Alhait - In charge of implementing the ML suite. This includes taking data from the camera polling module and running the face detection and tracking algorithms on each face in the frame. Will be in charge of developing the neural network model along with dimensionality reduction model to process each face that is detected and outputting a feature vector. This vector is then compared with the vectors of each resident and the verification result is sent to the main software module. A stretch goal is to superimpose the camera image with an IR image to prevent printed images without a heat signature from bypassing the system security.

All group members will help upon integration of each subsystem to the overall system and will serve as a resource for other members to bounce ideas and collaborate with.

C. Budget

Budget located in the Appendix of the report (Figure 3). By the time of writing the design report, we currently have approximately $245 left in the budget.

D. Risk Management

On the date of this writing, our three main risk include slow computation, inaccurate predictions, and network failures. There are five output devices and two input devices that all be controlled by one RasPi process. On the same RasPi, we will also be putting the face image into the neural network and running the testing suite in parallel to the hardware controlling modules. Lastly, our Pi will also be constantly listening for web requests sent by the client which then also have to be processed by the RasPi software. The server must be able to also send important updates to the webapp clients in response to real time events occurring in relating to the physical system. All of these tasks put together will be computationally expensive, which could cause us to not meet our time constraints given in previous sections. To mitigate this risk, we plan to implement threading on polling of camera images and plan to limit the amount of frames/time in which face captures are given to the ML test suite. In addition we plan to purchase a second Raspberry Pi if we find that the system cannot handle these operations given the optimizations we implement. The second Pi will be in charge of handling all server/client connections, which will allow the second Pi to focus on the ML suite and hardware control.

Our next risk is poor classification accuracy which could stem from factors including poor lighting, ineffective model

generation and poor facial detection/tracking. To mitigate the error that could be introduced with poor lighting, we plan to use a camera module equipped with IR flashlights, which will significantly brighten facial features in the dark. To mitigate the risk of a poor training model, we plan to compare our model with many others used for facial recognition. This will allow us to compare our classification results to make sure our algorithm works well enough on test data from the real world.

Since communication between the RasPi and the WebApp will be implemented using the server client HTTP requests, there is a high chance that packets sent over the web will be lost or corrupted in transit. To mitigate this we plan to implement a server client model that uses acknowledgements to verify that messages are being received correctly.[6]

In terms of schedule risks, a big risk is that we run into unforeseen problems when controlling the hardware and integrating the web app network capabilities. For this reason we are leaving more than two weeks of time for integration of all components of the system. We also plan to use distributed computing principles learned in other courses to maximize reliability. We have also implemented the keypad which serves as the fail safe for ht system if any of the backend systems crash.

VII. Related Work

Raspberry Pi Door Strike Access Control System With Android Tablet by MakeabilityLab [7]

Face recognition with OpenCV, Python, and deep learning by Adrian Rosebrock [8]

VIII. Summary

Was your system able to meet the design specifications ? Describe very briefly the limits on your system's performance, and any obvious things you can do to improve the system performance if you had more time.

TBD

1. *MVP*

Our MVP will consist of of the door security system that is able to unlock using face detection. The system will record a log of events that occur within the system and will send log events to the client. Client will be able to remote unlock the door and monitor system status. Client will also be able to add new users to training model. RasPi will control speakers, LED strip and electric strike. The sensors that the RasPi receives data from includes the keypad, ultrasonic sensors, and 5MP CSI connected camera. These components will be fixed to a door and frame which will emulate real life conditions.

2. *Stretch Goals*

Provided below is a list of stretch goals:

- Implement live stream to see/hear events at the door in real time
- Implement PA system in which resident can remotely speak with door visitor
- Superimpose IR camera feed with regular camera feed to prevent photos from being used to trick the system
- Adding notifications that include image/video files captured from RasPi
- 2 Camera 3D profile of person at door

References

[1] Lecture on Facial Detection and Tracking, https://www.youtube.com/watch?v=WfdYYNamHZ8

[2] Facial Detection with OpenCV, https://www.datacamp.com/community/tutorials/face-detection-python-opencv

[3] Steps for Facial Recognition, https://www.worldscientific.com/doi/suppl/10.1142/p155/suppl_file/p155_chap01.pdf

[4] FaceNet: A Unified Embedding for Face Recognition and Clustering Florian Schroff, Dmitry Kalenichenko, James Philbin

[5] Foundations of Machine Learning, Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar, 2018

[6] http://packetlife.net/blog/2010/jun/7/understanding-tcp-sequence-acknowledgment-numbers/

[7] https://www.instructables.com/id/Raspberry-Pi-Door-Strike-Access-Control-System-Wit/

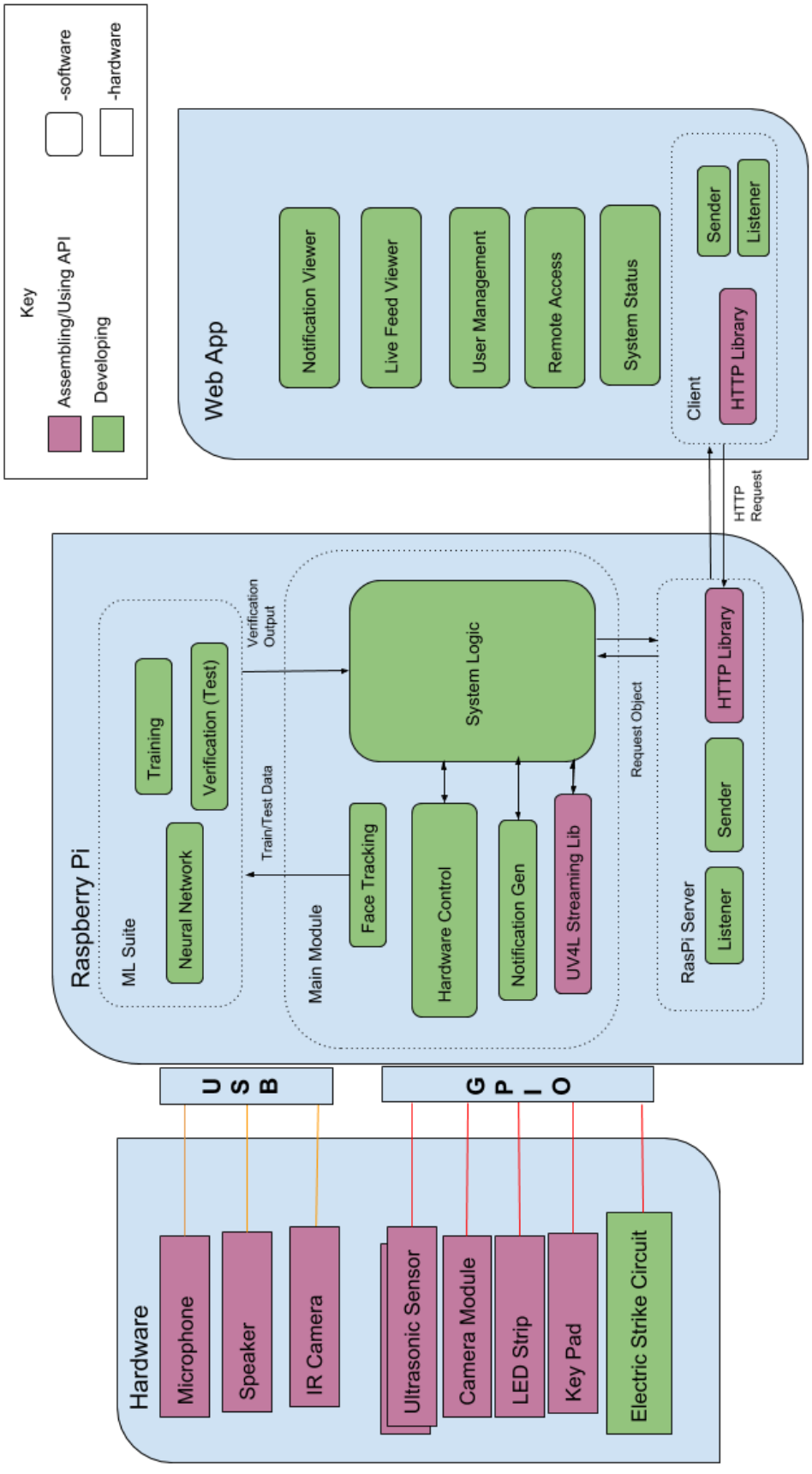
[8] https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/

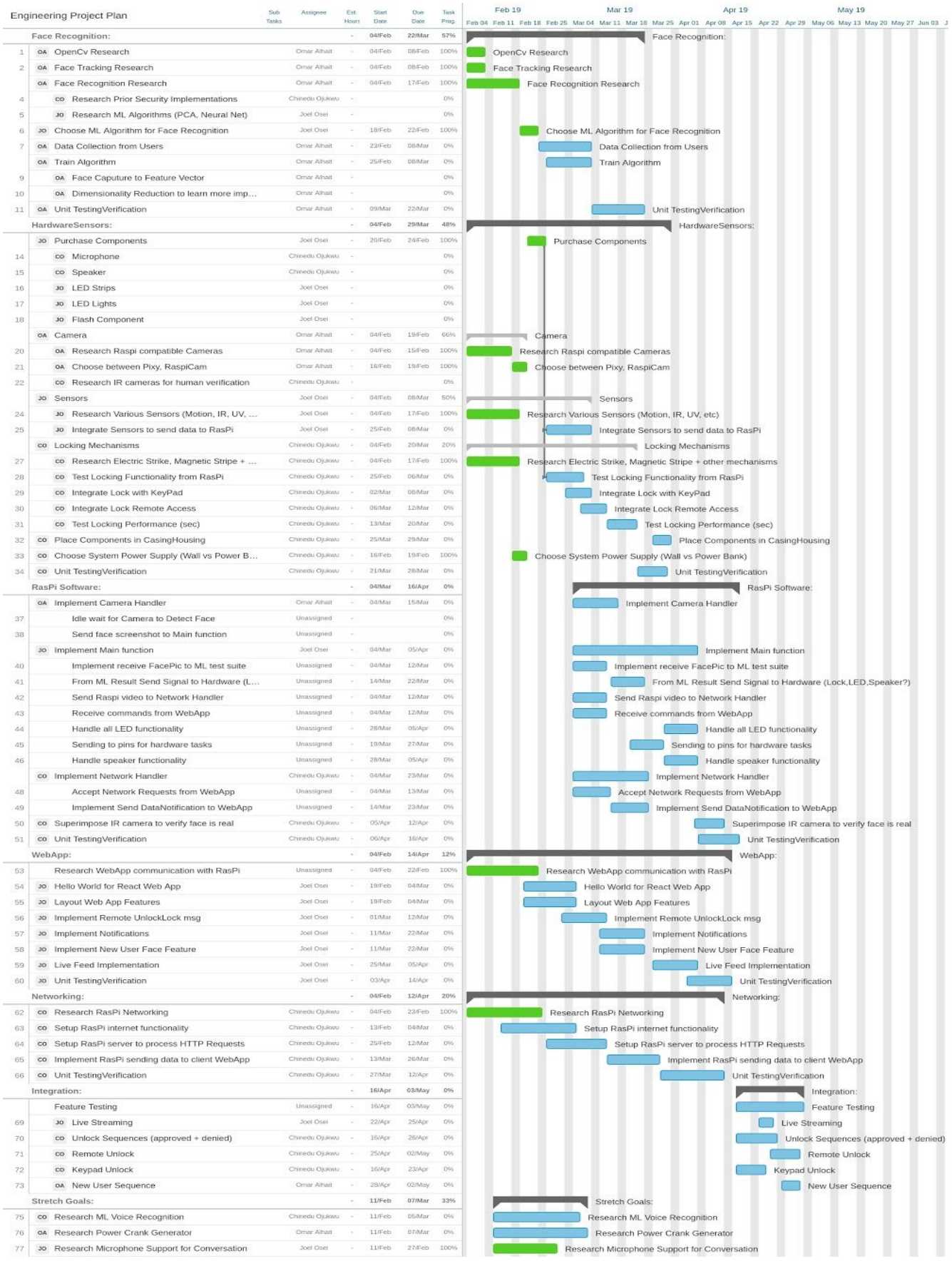
[9] https://quintagroup.com/cms/python/flask

[10] Steps of Facial Recognition, https://www.worldscientific.com/doi/suppl/10.1142/p155/suppl_file/p155_chap01.pdf

[11]

Appendix Figure 1. [Block Diagram]





Appendix Figure 2. [Gantt Chart/Milestones]

Item Name	**Cost**
Raspberry Pi v3 Model B+ x2	70.00
HES 500 Electric Strike	89.97
Flex Cable for RasPi Camera or Display - 1m	3.95
Flex Cable for RasPi Camera or Display - 2m	5.95
Adafruit CSI or DSI Cable Extender x2	5.90
5v Relay Module	6.99
5V Power Adapter x2	15.00
LED Strip	24.95
UltraSonic Sensor	3.95
Numeric Keypad	7.50
Mini USB Microphone	5.95
GPIO Breakout	7.95
Amplifier Breakout	5.95
3 Ohm Speaker x2	3.90
Wooden Demo door	69.00
Schlage Cylindrical Knob	21.20
12V DC Power Adapter Supply	5.00
Total	~355.00
BUDGET REMAIN	~245.00

Appendix Figure 3. [Budget & Parts List]