Implementing an Autonomous Raspbian Security System with Mobile Application

Patrick Yambao, Wayne Pham
Indiana University – Purdue University Indianapolis
CSCI 43300 Intro to Internet of Things
Professor Yao Liang
Teaching Assistant Hongyan Zhou

Abstract

The development of various IoT devices has advanced rapidly in recent decades. This project aims to develop an application that will function similarly to a home security system by utilizing raspberry pi four and the many sensors that comes along with it. Many revisions were done in the final project proposal that didn't make it in the final product, such as the DHT11 sensor, the keypad matrix, the smoke sensor module, and the ultrasonic sensor module. The final application includes sensors such as the LCD screen, the raspberry pi camera module, the PIR motion sensor module, and the servo motor used to rotate the camera around. In addition, the system also includes a passive and active buzzer and an RGB led. The final security system became a security system that will be fully controlled remotely via keypresses of the user's keyboard and mobile application, Telegram.

Student Information

Patrick Yambao	Wayne Pham
Student ID: 2000390924	Student ID: 2000342053
Email: payambao@iu.edu	Email: wpham@iu.edu

1. Utilized Sensors

1.1. I2C LCD 1602

The 16x2 LCD is like a touchscreen; however, it is mainly best used to display short messages or information. The typical LCD usually contains 16 pins, but not all need to be used. This final project only implemented four pins: the VCC, SDA, SCL, and the GND in 4-bit mode. We were able to take advantage of the LCD screen by displaying important messages to let the user know that the security system is either active or inactive. It will also show a statement that an intruder has been detected. For the LCD to fully function in our raspberry pi, we needed to configure the raspberry pi interfacing options and enable the I2C port.

Initially, we wanted to use the LCD to integrate some password security systems using the Keypad Matrix to enter some passwords or enable/disable the security system. However, due to the inability to fully interface both the LCD and Keypad Matrix, we decided to use the pygame module to control the system via our keyboards fully.

1.2. Infrared Motion Sensor (PIR)

The primary purpose of our security system is to detect some movement and alert the user that there is an anomaly saw and proceed to capture video or photos of the anomaly and send the attachment to the user's email or phone. With that, we utilized the PIR sensor to detect an object or organism above absolute zero temperature since they emit infrared radiation. The infrared wavelengths are only detectable by electronics which is where the PIR sensor comes in very handy. Typically, the PIR sensor will detect any object that disturbs any disturbance within the infrared signature of the room. Therefore, it is a susceptible sensor that can repeatedly detect movement within its field of vision; however, the PIR sensor can be configured for its sensitivity and the time it sees something by adjusting the potentiometers behind the sensor. Like the LCD, the PIR sensor is very straightforward to use. To connect it to the raspberry pi, we just needed to connect the three pins within the sensor to a 5V, any GPIO pin, and a ground pin.

1.3. Raspberry Pi Camera Module

The Raspberry Pi Camera module is an essential part of this final project. It is the primary way the pi will communicate to the user and warn the user that a foreign entity is near the device. Though at first, the camera module looks intimidating, the initial setup of the camera module is very simplistic. First, we only needed to attach it to the raspberry pi'sdesignated spot. Then, we needed to configure the raspberry pi interfacing options by enabling the camera.

1.4. Servo Motor

The servo motor is a compact package that consists of a DC motor and sensor that can rotate in a 180-degree range of motion via their horn. To connect the servo motor in the raspberry pi, we will need only three pins again for the 5V, any GPIO pin, and the ground pin. The angle of the motor is set up along the length of the pulse; therefore, PWM is particularly useful in controlling the motor. We utilized this motor by potentially attaching the horn with the camera and can rotate the camera to the left or the right via keypress. We now have a fully functioning system that we can control in the direction we want.

2. Other Components

2.1. RGB LED

As the title suggests, we implemented an RGB led to our system fully expanding from project one by using a much more complex led that we can utilize by changing its colors to signify something meaningful. The initial setup of the RGB led only requires three 220Ω resistors and 4 GPIO pins, with one of the pins being connected to a 3.3V. We utilized this led by having multiple functions that will change the color of the led. For example, we have the functions purple(), red(), and lightBlue(), each corresponding to the current state of the security systems.

2.2. Buzzers

Passive

The primary idea behind the passive buzzer is using the buzzer to play a tune to signify to the intruder that they have been detected. In addition, we were able to play around with the passive buzzer more by having it play a particular tune or song. The difference between the active and passive buzzers is that passive buzzers don't have an internal oscillator; therefore, we need a pulsating AC signal to generate a tone.

Active

We utilized the active buzzer by generating a generic beeping sound whenever the system is armed and deactivated. Both the active and passive buzzers were connected the same way: two pins connected to any gpio pins and the other connected to a ground pin.

3. Result Analysis

3.1. Code Development

We will be using Python 3 and the Thonny Python IDE inside raspberry pi to code all the essential files we needed to create the finalized product that we wanted. For the LCD screen portion, we found a library that we could use for the full functionality of the display. Coding the PIR sensor is essentially simple in a way that we didn't need to import any libraries to use the sensor besides the usual imports that we usually did. The rest of the sensors and the other components were essentially direct. The only other sensor that was a bit complex was the raspberry pi camera and the servo motor since the raspberry pi camera needed to have the pi camera module imported. The main functionalities that we used for the sensors and the other components were the main methods found in many tutorials online. We could reference the many codes that were provided free to use and revise. We were able to manipulate it to create a functioning autonomous security system.

Lastly, to further clean the code and make it easily accessible, we divided the main code into several files that we then imported into the main file. This will allow Wayne and me to configure the code quickly and see which parts are working and which are not; thus, debugging the entire system was a lot easier and more efficient.

3.2. Working Application

The main idea behind our security system is the ability to detect any foreign entity that is closed within the PIR field of vision and alert the user that there is an intruder. Due to our inability to fully implement both the Keypad Matrix and the LCD screen to function altogether and create a working password system. Essentially the way it will work is that the primary user can press a specific key in the keyboard, and they will be able to control the system.

For example, if I press the keyboard "a," the LCD screen will display the message that the security system is turned off, make the RGB turn red, and then make a general beeping sound. The same way will happen if I press the keyboard "b," activating the system and turning blue. If I want to start the security system and see any intruders, I can press the key "x" and activate both the PIR sensor and the camera sensor and see any intruders. On top of that, if I want a specific direction the camera wants to face, I can rotate the camera via the left and right arrows on the keyboard, and the servo motor will turn the camera accordingly.

3.3. Motion Detected

What happens if the motion is detected? First, suppose the PIR sensor detects the movement. In that case, we have the camera sensor module configured to take photos of the intruder and send those photos in the user's email with an attachment stating that the PIR sensor has captured an anomaly. Once motion is detected, the passive buzzer will generate a tune telling the monster that the system has detected it. After that, the system will return to its original state, and it is up to the user to decide if they want to activate the system again.

4. Bonus Attempt

4.1. Mobile Application

As stated in the final project requirements, an additional ten-point bonus will be given to those who show exceptional work for the project. With that, the extraordinary work that I will be showcasing is using a mobile application found in both the iOS and Android play store called *Telegram*. Similar to *WhatsApp*, Telegram is a messaging app, but in June of 2015, Telegram published the Bot API, which will enable both humans and machines to use the platform. The initial setup of the application is quite simple: downloading the app from the store and signing up with your mobile phone number. From there, I needed to add a bot called BotFather which essentially implies that this bot is the father of all bots that accept special commands and can create bots. For the raspberry pi to connect to this application, I needed to create a bot under BotFather, which came with a specific token that I can use for the raspberry pi to communicate with the app. From there, I just needed to import telepot in my code and make sure I have telepot installed in the raspberry pi by doing *sudo pip install telepot*.

5. Issues

5.1. PIR Sensor

The main issue with the PIR sensor was that it was too sensitive to use. In other words, the PIR sensor was initially set up to detect motion; however, once the PIR sensor detects motion, it will continuously see motion and snap pictures which will then bombard my emails with photos taken from the Pi camera module. We could get around this by revising the while loop into a range loop. Instead of continuously monitoring the system, we decreased it into six iterations. However, this problem can also be fixed by implementing multithreading and having two functions constantly running simultaneously. Still, we just went with the easy route and did in a range of 6, and the user can re-arm the system with a simple keypress.

5.2. Keypad Matrix and LCD1602

This was the main problem that we spent hours trying to fix. To put it simply, we could have both the keypad matrix and LCD screen working. In

addition, we were able to integrate the keypad matrix to work in the LCD screen; however, it was generating the python code to have a password system security that became the tricky part. We ran into numerous errors, such as an infinite loop and the keypad matrix not registering when we tried to revise it. Therefore, we did not fix this problem; instead, we found an alternative: the pygame module that we can fully take advantage of with the keyboard presses.

5.3. File Overloading

The images and video the camera module captures will usually store the jpg picture and the mp4 video inside the same directory or file of the project. This became a problem since when we tied to do some system testing, the project folder will just get flooded with jpg and mp4 attachments. To solve this, we revised the code so that whenever the PIR sensor detects motion, the captured photo or video will be stored inside a folder of the same directory of the project called *Database*.

6. Team Contribution and References

6.1. Team Contribution

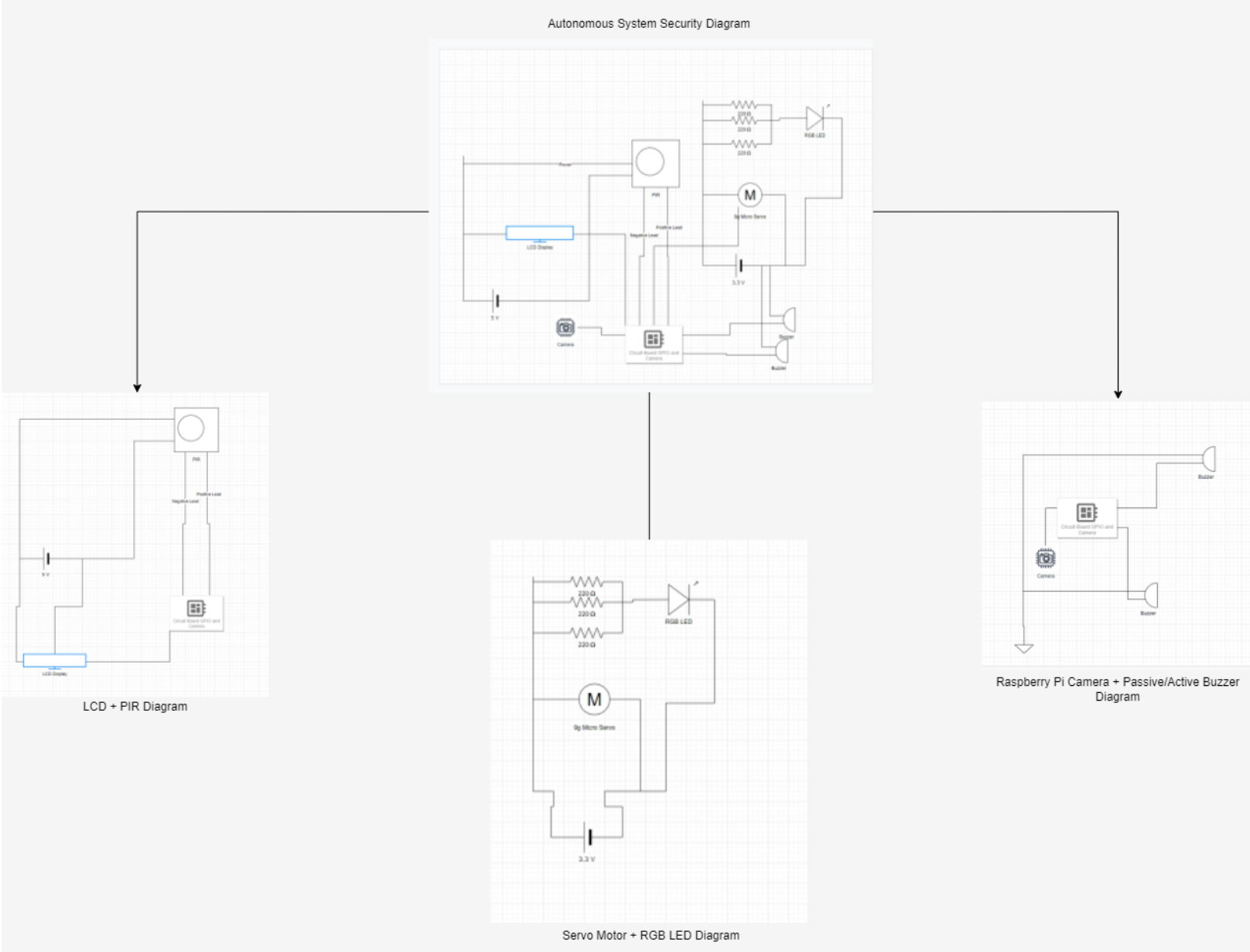
Wayne

In charge of the sensors and the circuit diagrams. As well as the LCD and keypad matrix portion that we tried to initialize at the beginning. Helped contribute to the coding part of several files and helped polish up the report.

Patrick

In charge of setting up the main project in his raspberry pi and writing up the report. He contributed to generating the rgb, PIR, camera, and servo code.

7. Circuit Diagrams



8. References

Doe, J., Godlewski, D., Felix, Futuristic_Paladin, Asim, Generationxcode, Tim, Ismail, Henry, Peter, Delivery, O., Football, X. Y., & Champs, C. (2021, January 25). *Raspberry Pi Servo Motor Control*. Raspberry Pi Tutorials. Retrieved December 11, 2021, from https://tutorials-raspberrypi.com/raspberry-pi-servo-motor-control/.

How to setup an LCD on the raspberry pi and program it with python. Circuit Basics. (2021, November 14). Retrieved December 11, 2021, from https://www.circuitbasics.com/raspberry-pi-lcd-set-up-and-programming-in-python/.

How to use buzzers on the Raspberry Pi. Circuit Basics. (2021, November 29). Retrieved December 11, 2021, from https://www.circuitbasics.com/how-to-use-buzzers-with-raspberry-pi/.

Motion detection video captured email alert using Raspberry Pi 4. Robotica DIY. (2020, April 26). Retrieved December 11, 2021, from https://roboticadiy.com/motion-detection-video-captured-email-alert-using-raspberry-pi-4/.

NickL17, & Instructables. (2017, October 5). *Set up telegram bot on Raspberry Pi*. Instructables. Retrieved December 11, 2021, from https://www.instructables.com/Set-up-Telegram-Bot-on-Raspberry-Pi/.