Final Presentation Transcript

Slide 1: Group Intro

NOAH – Hello Everyone! Welcome to our Senior Project Final Presentation! Our Group created a Fire Detection System and the members of the group include Saul, Jeric, Chris and myself (Noah).

Slide 2: Introduction

The main drive of this project started back in the early months of the Fall 2020 semester with the emergence of the devasting wildfires throughout the world, especially in California and Australia. In response to the millions of acres burned in 2020, our group decided to be proactive and create a fire detection system. The main issue is a lack of early fire detection for the homeowners living in remote areas or have large open land. The goal was to create a consumer accessible product that could be implemented on these homeowners' property. The main components of this project include a camera on a rotating servo that transmits video to the object detection software, along with sensors that detect low soil moisture, and a buzzer to alert the homeowner. This was completed by taking advantage of many open-source programs such as LabelImg, Darknet, Python, Raspberry Pi, and Arduino to achieve these results.

Slide 3: Market Research

Now getting into the Market Research. The main objectives of this project are increasing safety and insuring conservation of the environment, however, that does not mean we are immune to environmental, ethical, political, and social constraints. Due to the technology being included. Some potential ethical/political constraints that could cause an issue include the use of a camera with object detection due to the ongoing debate about privacy. However, the intended location of this system would be on private property with the owner's permission. False alarms can be a potential social issue, so we want to ensure detection accuracy reaches a certain threshold before alerting the user of a fire. Now Chris will go over the project plan.

Slide 4: Project Plan

CHRIS – Hello I'm Chris Gripkey, an Electrical Engineering student working on this project. I oversaw image annotation, methodology and troubleshooting. Here is our original project plan. We designed this project to have to data inputs, pictured as green orthogons, to feed camera and humidity data to the microprocessor. After running the visual data through an implementation of YOLOv4 Tiny. The resulting probability, if above the threshold, would trigger an alarm. If not, the humidity data would be accessed and trigger a simple warning if enabled and loop back to repeat the process.

Slide 5: LabelIMG Fire

When classifying images for the convolutional neural network (or CNN for short) we chose to minimize the number of classifications to three things: fire, smoke and nofire. The reason for this was to include fire-like adversarial images as nofire classifications to delineate fire from other sources that typically could be classified as potential fire sources. We also included smoke as an indicative classifier to reinforcement of fire confidence.

Slide 6: LabelIMG Other Labels

Here you can see some examples of the annotation of the adversarial images to better train the CNN. Due to hue prominence and saturation. a primarily red room and extreme contrast light sources were needed such as those pictured. In total we annotated 565 images with about ~60% fire images and ~40% non-fire.

Slide 7: YOLOV4 Object Detection

To better understand our results, we looked at the confusion matrix and accuracy of the model. Unfortunately, due to the need of using a faster, more lightweight CNN for evaluating the camera input, the overall average precision of the trained CNN we chose to use was only 86%. This rating is too low for real world applications, where a one in a million false positive rate is expected for utilization. We will need to improve upon this rate in future iterations. Jeric will explain the deployment next.

Slide 8: YOLOV4 Object Detection

JERIC - Hello, I'm Jeric, and my work on this project was focused on the training and deployment of the neural network.

We had tried different levels of training as our dataset grew larger. From the chart that you see here, this is a model that was run for a total of 10000 iterations compared to the recommended 6000 iterations per 3 classes. As Chris mentioned, we decided to use a model run for less iterations and with a lower resultant precision from training. The problem with running with more iterations causes a higher chance of false positives due to overtraining the most represented class in the image dataset. I was able to train locally on my machine taking about 2 hours per model. This short video demonstrates the model detects fire from a lighter when run from my desktop and webcam. (4 second video)

Slide 9: Deployment of Neural Network

Despite using the tiny model, the Raspberry Pi is not capable in processing it. To work around this, the model is converted and optimized to Tencent's ncnn framework. This framework is designed for use in processing performance found in mobile phones.

The code that you see on these slides are derived from the official NCNN repository examples. It is in C++ from an existing project by Q-Engineering in the Netherlands. net.h is a header called from the ncnn library, and a few OpenCV libraries are added for handling inputs. The modifications I made here are in the class names we will be detecting with this model: fire, smoke, and nofire. The next functions declared, which are not pictured, are the object detection and labelled box drawing functions.

Slide 10: Deployment of Neural Network

Here in the main function, the trained model that was converted from yolo to ncnn format is loaded. Originally, this program was set to ask for an input JPG file, but I modified it so that it will continuously capture frames from the default camera index until the user hits Q from the camera window to exit the program.

Slide 11: Deployment of Neural Network

This video is a demonstration of that code in runtime. The camera is a standard Raspicam that can take images up to 1080p, but when run as a video in this resolution, the frames are processed much slower, and the camera signal becomes garbled and discolored into a bluish color space that does not detect fires. On the other hand, lowering the resolution causes decreased detection and accuracy, so running at 720p was the only passable option for detection. You can see from the video that a fire, represented by a NULL class, only appears for a couple of frames for the whole video. As mentioned, the camera is mounted on a servo which Noah will now go over.

Slide 12: Servo

NOAH - Hi it's Noah again. I am going to talk about the part I focused on which was the servo and the camera mount. The wiring of the servo consisted of signal wire going to pin 11 which is a GPIO pin necessary to send a PWM signal to the servo then we have our ground going to pin 6 and 5V to pin 4. We went with a continuous rotation servo as a regular micro servo would not be sufficient. We took the FOV of the Raspicam, which was ~62 degrees, to determine that is necessary to have 6 different positions of the camera to cover a full 360 degrees. As show in the diagram in the bottom right corner we have 6 different camera stop positions 60 degrees apart. As I started researching the programming of the servo, I realized that it did not have a potentiometer which is a key component, to allow the servo to recognize its position. I had to work around this by changing the duty cycle and using a protractor to get the approximate angle within 2-3 degrees of the exact degree. Here's a short demo of how the servo works. (video)

Slide 13: Servo code

Let's briefly go through the important parts of the code. I set pin 11 as my output pin and gave it a 50Hz which is the input frequency the servo requires. Then I started my first while loop for the counterclockwise rotation for 6 stop positions. Finally, another while loop to stop at 6 positions in the clockwise direction.

Slide 14: Resulting Camera View

Here is a quick demonstration of the camera view of the servo running for the first set of rotations. (video) Saul will now go over the Wireless Sensor Network.

Slide 15: **Wireless Sensor Network**

SAUL – Hello my name is Saul and I primarily worked on the Wireless sensor Network.

Slide 16: OVERVIEW

So, a general overview of the system will consist of a “central” node where all other sensor nodes will transmit to. As you can see once the data has been gathered by the moisture sensor, it then proceeds to transmit the data to the central node. The central node will then display the reading to the LCD screen and will check if it is above a certain threshold. In this case 20%. If it falls below this set threshold then it will trigger the alarm but can be silenced using a button.

Slide 17: Parts

In total, we used 7 components to construct this system, 5 for the central node, and 4 for the sensor node. The central node was constructed from the Arduino microcontroller, transceiver, LCD Screen, Buzzer, and button. Moreover, the sensor unit was constructed from similar components such as the Arduino Microcontroller and transceiver but also included a battery and moisture sensor. The total cost for the central node and one sensor node came to a total of about 100 dollars.

Slide 18: Central Node

The central nodes main features are the following. The Central module can be powered by any plug-in outlet in your home in conjunction with a 5V AC adapter and a USB to mini-B port. As previously mentioned, it displays moisture levels from surrounding nodes and below a set threshold level triggers an alarm. In addition, a button to silence the alarm. As you can see on the right-hand side you can see a view of the central node.

Slide 19: Wiring

Here is the overall schematic for the central node and how all the different modules were connected. An important connection to mention would be the Inter – Integrated (I2C) PINS highlighted in red. This form of communication protocol enables us to use the LCD with significantly less GPIO pins than a traditional LCD would have.

Slide 20: Sensor Node

The sensor nodes main feature is that it able to transmit up to 135m long. This was done in a residential setting so the range could be higher in an open setting where there are no obstructions such as other houses. As previously mentioned, it transmits every 6hrs and can run up to 11days. The battery can then be recharged using a micro-USB cable.

As depicted in the image you can see an aerial view of what 135m looks like. Particularly from the Walter stern library from the CSUB campus.

Slide 21: Wiring

Here is the overall schematic of how the pins were connected to the Arduino microcontroller. An important connection to mention would be PINS D11 and D12 which they are the assigned SPI pins on the microcontroller. SPI stands for Serial Peripheral Interface—it's a de facto synchronous communication bus standard, it boasts both simple implementation and high-speed data transfer capability. Here we are using it to transmit data to the transceiver so it can then be transmitted to the central node.

Slide 22: Antenna

The antenna was constructed using a strand of solid core wire. For the antenna to be functional we created what is referred to as a quarter whip antenna. To determine the length of the antenna calculations needed to be made. The length of the wire is equal to ¼ of the wavelength. therefore, if we look at the equation for wavelength, we can calculate it using the speed of light which is 3x10 m/s and the frequency being used for transmission which 433MHz. Evaluating, this then gives us a result of 17.3 cm.

To the right you can see the antenna soldered on the transceiver breakout board.

Slide 23: Code

So, for our system, one library used was RadioHead which was created by Mike McCauley. As you can see it provides a lot of functionality for setting up the transceiver and implementing it to your will. The second library we used was LiquidCrystal_I2C which was obtained through johnrickman on GitHub. This library needed to be used due the LCD screen using I2C serial communication protocol but is very similar to the Arduino LiquidCrystal library.

Slide 24: Continued

One aspect that really needed to be addressed with transmitting and receiving was converting the moisture sensor reading from the sensor into a format that would be transmitted successfully. Using this library, transmission occurs in packetized bits,

therefore we needed to break the sensor reading value into single character bits which could then be transmitted and received correctly.

Slide 25: Demo

Now the Demo. Before we play video, I would like to provide some context on what you will be seeing. So, the total distance between where the central unit and the node is about 15m or so. Here in the video, I will be taking and transmitting the moisture level every 3 seconds, but as mentioned before the final product will be taking moisture levels every 6 hours. The threshold value will be set at 20% in accordance with the final products threshold level. Starting the video, you will see the sensor embedded in very dry soil, therefore the alarm will sound off. I will then proceed to continuously add water to increase soil moisture levels. I will also be continuously pushing the button to turn off the alarm after each reading until it exceeds the threshold value for the alarm.

Chris will now address the timeline of the project.

Slide 26: Original Timeline

CHRIS - Hello Chris Gripkey here. This is the original timeline for our project. We did complete all but one task on the timeline, however the duration of each task varied from our original expectations. We did not account for several setbacks including several instances of hardware failure and a delay on funding. As a result, to meet timeline milestones, we were forced to compromise on software and hardware used due to resource availability and timing. Examples include using YOLO instead of our own CNN, images without IR spectrum for the training and forgoing the development of an external chassis for the camera.

Slide 27: Project Challenges

This project experienced several challenges, several of which can be tied to COVID restrictions. Until the entire group was vaccinated, we were unable to meet. Due to the ambitious nature of this project, the scale and shortfalls of the project were not fully accounted for. We had multiple pieces of hardware that were faulty or not up to specification, which added lead time or not having the appropriate part due to funds. Another difficulty faced was the limitations of the Raspberry Pi. Running programs simultaneously on the Raspberry Pi also affected the consistency of the servo rotation as well as the speed of the object detection program.

Slide 28: Project Learnings and Plans

The project was very educational when coming to terms with expectations and collaboration in a virtual setting. Microsoft Teams made it possible to work remotely for the full course of the project. Teams is incredibly flexible for collaboration combining filesharing, chat client, and task/deadline setting.
For future iterations we plan to add an IR sensor and use a camera with a normal

recording space and a focusable lens to due to the incidental bleed caused by Infer-red radiation. Add additional annotated images to the training library to further reinforce the CNN accuracy. Using a 360-degree servo with feedback to accurately determine camera position for recording and adding a second servo to allow y axis rotation. Additionally, implementing a specialized CNN and moving from Raspberry Pi OS to onboard implementation to improve performance.

Slide 29: Conclusion

JERIC – To wrap everything up, it is possible to have an implementation in the initially proposed scheme. The processing speed can be improved with the use of a different single board computer with an onboard GPU, or a neural processor stick added onto the Raspberry Pi. As Chris mentioned, we need to refine the neural network for the object detection to be passable as a safety tool and for the purpose originally intended. Lastly, consolidating the code and integrating the wireless sensor network will be necessary to utilize each component to raise the same alarm and make this project more complete. Overall, it was an enriching experience attempting to provide a different solution to a complex problem. We thank you for your time listening to our presentation.