Zane Smith

Final Report: Security Camera and Facial Recognition Systems

Report Overview

This paper documents the completion of a two-part project aimed at enhancing the security of multiple buildings within a school district. To provide focus and clarity, the paper will be divided into two sections, the first detailing the installation and configuration of a Power over Ethernet (POE) security camera system, and the latter covering the development of a facial recognition system using a Raspberry Pi. In addition to an introduction, each section will discuss objectives, an overview of the implementation, the challenges faced and how they were overcome, the relation to Computer Science degree, and final outcomes. This report will demonstrate how the combination of these projects has given me the opportunity to apply and further enhance the wide range of technical skills and concepts learned while obtaining my degree in Computer Science.

.

Security Camera System

Introduction and Objective

The first portion of this project focused on the implementation and configuration of a POE security camera system designed to enhance the overall security of multiple buildings with a school district. This project became urgent upon one of the buildings reaching the final stages of a major renovation. Many of the existing cameras were inadequate, as they were either outdated or in poor locations and provided minimal coverage following the renovation. Additionally, the renovation included an expansion, which required the installation of new cameras.

The project was carried out in multiple stages, including planning, hardware installation, network configuration, and system testing. The primary objective was to design and implement a modern, reliable, and secure camera network that would enhance monitoring capabilities while integrating seamlessly into the school's existing IT infrastructure. Furthermore, the project prioritized security considerations and placed an additional emphasis on achieving optimal functionality and coverage. The project also included the installation and configuration of access points that followed a very similar process.

The following section outlines the goals, processes, and outcomes of the security camera installation, highlighting how key networking and hardware concepts were applied to meet the project's requirements and objectives.

Implementation Overview

The implementation of the POE security camera system involved a plethora of carefully planned steps to ensure seamless deployment. Each stage required a combination of technical expertise, problem-solving, and collaboration to address challenges and achieve the desired outcomes. I worked closely and was assisted by my supervisor and one of the company's engineers throughout this process.

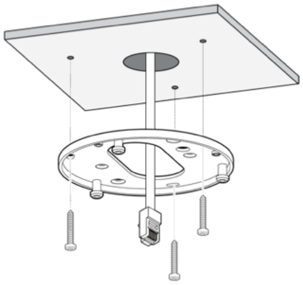
1. *Planning and Preparation*

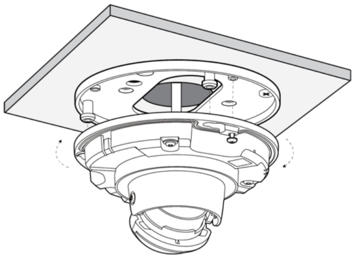
The initial phase of this project focused on identifying the optimal locations for the cameras we would be installing. While electricians had already run Ethernet wiring to general areas, a more precise plan was required before ordering cameras. The intended goal of this was to use the fewest number of cameras possible while still maintain optimal coverage. To achieve this goal, we arranged a meeting with principals, building administrators, and others that would have access to the cameras. This meeting was centered around the most critical camera locations, including entrances, high traffic areas, near rooms with valuable equipment, and areas containing sensitive documents. After determining the most essential locations and what they needed to cover, we then mapped out the location of the remaining Ethernet drops and the exact location/position that cameras would be placed for optimal coverage.

The next portion of planning was acquiring equipment, primarily including Cat 6 patch cables, cameras, and camera mounts for outdoor locations. Several different camera types were chosen for the project, all manufactured by Verkada. This choice was made due to many advanced security features, analytics capabilities, remote accessibility, and streamlined management. More specifically, the cameras selected for this project were from the Verkada Dome Series (both the CD22 and CD63-E) and from the Verkada Fisheye Series (CF81-E).

2. *Hardware Installation & Configuration*

The physical installation was a relatively straightforward process and began with mounting the cameras. This consisted of removing a ceiling tile, drilling holes in tile, and then securing the provided mounts. Once the mounts were in place, we threaded the Ethernet cables through the mounts, attached the cameras to the mount, and plugged the cables into the cameras' RJ45 ports. Below are basic diagrams showcasing hardware installation.





This step also involved network and port configuration. Initially, we enabled the ports and ensured they were active. For security purposes, the schools network blocks any unauthorized devices. To address this, we implemented MAC authentication bypass by whitelisting each camera's MAC address for its specific port. For additional security, we configured Virtual Local Area Networks (VLANs) to isolate camera traffic from other network devices and to reduce the risk of unauthorized camera access. Additionally, we whitelisted Verkada domains on the school's firewall to ensure seamless communication. Following this, we located the intermediate distribution frame (IDF) associated with the camera's location and connected the patch cables from the patch panels to the switches, very similar to the image shown below. We ensured the cameras were receiving power, by the camera's LED indicators.

To continue setup, we used Verkada Command, Verkada's cloud-based platform that allows users to monitor and manage their devices. Cameras were added to this platform using their serial numbers. Additionally, the cameras required a Dynamic Host Configuration Protocol (DCHP) enabled network to obtain an IP address upon setup. Once the cameras we configured and online, we were able to assign a static IP address to ensure consistent network communication and to simplify troubleshooting making the cameras easier to identify. To finalize this step, Verkada Command was used to assign unique names, locations, and configure various settings such as orientation, bandwidth, and quality.

3. *Testing and Optimization*

After installation and configuration, we tested each camera, checking for blind spots, ensuring proper coverage, and making necessary alignment adjustments. The caps of the cameras were screwed on using the provided security screws and screwdrivers. Network configurations were reviewed to ensure proper configuration and firmware updates were applied to all cameras. Finally, network monitoring tools were used to evaluate performance and to check for any connectivity issues.

Challenges and Solutions

1. **Challenge**: In some cases, cameras failed to power on.

   Solution: This was the most common issue, and it arose due Ethernet drops being unlabeled or mislabeled leading to patch cables being plugged into the wrong port on the network switch. To resolve the issue, we used a toner tool to trace the cable. This process involved plugging one end of the toner into the cable at the camera's location and running the toner wand over the switch ports until the correct port was identified. In a few cases, we had to trace the entire Ethernet run through the ceiling to find the associated IDF. Once located, the cable was moved to the correct port, restoring power to the camera.

2. ***Challenge*:** Upon booting, some cameras were displaying a flashing blue LED, indicating camera was recording but not connecting to the server.

   Solution: The first troubleshooting step was to confirm the camera was receiving a local IP address from the DHCP server with the MAC addresses from the camera. If no IP address was assigned, we rechecked and fixed the configuration of the camera's port and the network settings.

3. **Challenge**: One camera was connected but took an unusually long time to come online.

   Solution: After Investigation, we found that the camera was assigned an IP address and had network connectivity, however, it was connected to an unmanaged switch. We determined which network run was connected to the unmanaged switch and disconnected the devices to isolate the camera's cable. The camera was then moved to the correct managed switch, which had the necessary configurations for proper operation.

4. **Challenge:** Several ceiling Ethernet runs were too short to reach the desired camera locations.
   Solution: New lengths of Cat 6 cable were cut and terminated using RJ45 connectors.
5. **Challenge:** Some cameras were assigned to the wrong VLAN, causing issues with connectivity and security. Additionally, some cameras experienced bouncing IP addresses due to misconfigurations.

   Solution: We double-checked the assignments for each camera, correcting if needed. Static IP addresses were reassigned as needed to prevent address conflicts. Any adjustments made were tested to ensure all cameras communicated reliably with the server.

Degree Relevance & Final Outcomes

The installation and configuration of the security camera system provided a valuable opportunity to apply foundational knowledge and technical skills gained throughout my Computer Science degree program. Various parts of this project required troubleshooting steps that mirrored the systematic methodologies that have been taught and refined throughout the computer science curriculum. Additionally, group projects and team-based assignments in my coursework prepared me for effective collaboration and communication with my supervisor, company engineers, and school administrators. Additionally, this project heavily relied on and reflected learnings in network fundamentals, security fundamentals, and system configuration. These concepts have been a core focus throughout the curriculum and directly relate to this project. The deepened background of hardware components the degree has granted me also proved useful throughout the project.

By the end of this project, we successfully implemented a modern and reliable security camera system that enhanced the security and surveillance capabilities within the school district. The project integrated effortlessly with the existing IT infrastructure, and improved network performance and security. Additionally, we were able to achieve all goals and desired outcomes of the project within the desired timeline. Furthermore, I was able to use this project to strengthen my understanding in various concepts discussed throughout the pursuit of the degree, as well as expand my knowledge and learn in areas beyond, yet still relevant to, the degree scope.

Facial Recognition System

Introduction and Objective

The second part of this project also focus on enhancing security through the use of cameras, however this portion concentrates on development. For this, I have designed and implemented a small-scale facial recognition system designed to run on a Raspberry Pi, using a camera module and Python. The facial recognition system uses computer vision and machine learning techniques to identify individuals from a live video stream. This project utilizes libraries such as OpenCV for image processing and Tkinter to create a simple and easy to use graphical user interface (GUI) to manage user interactions. The system captures and processes images of individuals, extracts facial features, and compares them with previously captured images to identify faces in real-time. If a match is found, the system identifies the person. To broaden the system's applicability, additional functionally was added to send out alerts if an unrecognized face is detected and serves as an automatic attendance tracker.

This project was carried out in multiple stages, including planning, development, and system testing. The primary objective was to create a reliable and cost-effective camera monitoring system to provide an addition layer of security within a school setting. The following section outlines the goals, processes, and outcomes of the project, highlighting how key computer science concepts were applied to meet the project's requirements and objectives.

Implementation Overview

The implementation of the facial recognition system was divided into similar stages as the security camera portion; however, it replaces hardware installation with software development. The initial key goal was to send out alerts when an unidentified person was found. Once that was functioning, I decided to expand upon it, creating an intuitive GUI so any staff member could easily use the system. Each day staff members send out an email to the secretary containing a list of absences in their respective room's. The lists get compiled and then emailed out to everyone. Since the system has the resources and everyone enters through the same doors, I decided to simplify this process by automating it.

1. *Planning and Preparation*

The first step included determining the scope and requirements of the project, conducting research, hardware selection, and choosing Python libraries. After research, I found most were using the Raspberry Pi 4b for facial recognition projects.

However, I selected the Raspberry Pi 5 due to the overall improvements made over the prior model that made it more powerful, which would be especially useful for computationally intensive tasks like facial recognition. I also selected an Arducam 5mp Camera, as it was a cost-effective alternative to the newest Raspberry Pi camera module. Additional hardware used for the project included a 64 GB micro-SD card, a power supply, a Mini HDMI to HDMI adapter, a keyboard, a mouse, and a monitor. Several key libraries were selected for the project. Of these, the most used were OpenCV for image processing, Tkinter for the GUI, and face-recognition for feature extraction and comparison.

Furthermore, there were several preparation steps taken before I could begin development. First, I had to download the Raspberry PI imager on my MacBook and use it to download Bookworm OS onto the micro-SD card. Following that, was light hardware assembly that included inserting the provided 22pin-22pin connector into the camera module, then into the Pi's CSI connector. Next, I setup a virtual environment to run the project without the risk of damaging the rest of Pi OS. Lastly, working in the virtual environment, I updated the Pi OS and necessary packages.

2. *Software development*

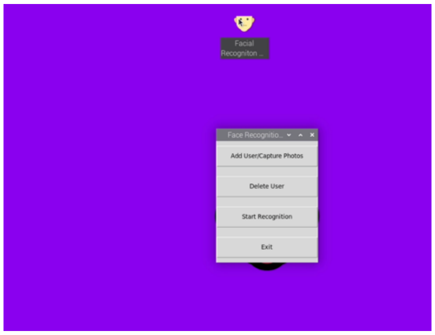
The code begins with the **image_capture.py**, used with OpenCV. This file is used to capture images and manage a dataset that stores captured images. The dataset is organized by the name of the person whose photo is being captured. After using image_capture.py, to save images, **model_training.py** was created. It works with the face-recognition library to extract facial encodings representing unique features and store them for future comparison. Next, **facial_recognition.py** was created to process and find frames containing faces from a live video stream. Faces found were compared with the stored encodings and the system identifies the person in real time.

Once these three files were created and working properly, the system was then expanded upon. Image_capture.py was adjusted so the camera window displays instructions on bottom of the screen along with the number of photos that have been captured, rather than outputting in the command line. Facial_recognition.py added the capability to send email alerts when an unidentified person is detected using the smtplib library, to establish a secure connection to a SMTP server. It also added the ability to automatically tracks attendance by tracking expected attendees (those that are in the system) and removing their names from the list if they are identified. This list is then emailed out once every morning. Afterwards, a GUI was created and added the functionality to remove users from the system. Lastly, a desktop shortcut was added to the Pi's home screen to simplify so much that anyone easily could use it.

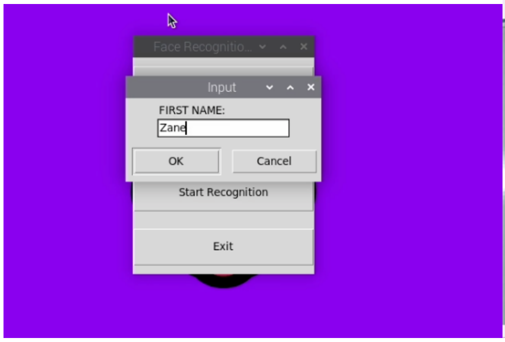
3. *Testing*

The last phase of this project was rigorously testing the system to ensure functionality and reliability. The known users we tested extensively to ensure accuracy and ensure minimal false positives or negatives occurred. It was also tested under different lighting conditions and user positions. Many edges case were accounted for throughout the testing phase. Below details a basic example of using the system.

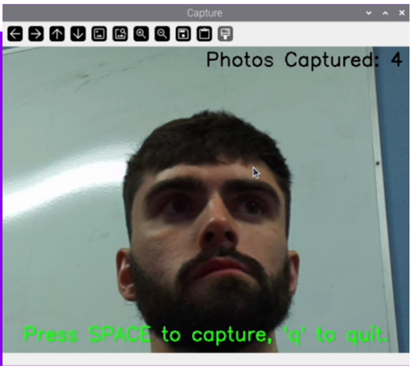
1. The user must first sign into the Raspberry Pi using their admin credentials. Once signed in, the user first clicks the desktop icon, and the main menu of the GUI will open, displaying 3 options.



2. If the user selects Add user/capture photos, they will be prompted to input the users first name, and another identical popup but for their last name will follow. After that, the camera window will open, and the user will have their photos captured and saved. Once they quit out of this window, model training automatically complete, and encodings will be updated and saved without the need for additional user interaction.



FIRST NAME: Zane

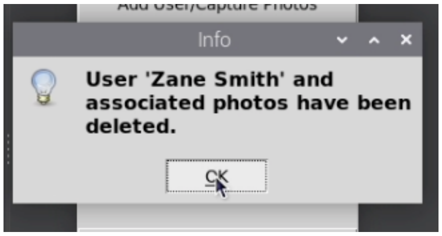


Photos Captured: 4

Press SPACE to capture, 'q' to quit.

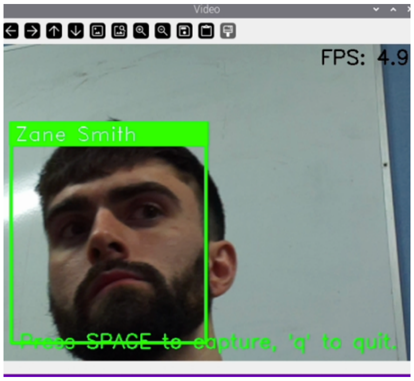
3. If the user opts to delete someone from the system, this will evoke the same input boxes for their first and last name as before. Once inputted, there will be a popup reading “Are You Sure You Want To Delete This User?”. If “Yes” is selected the user will be deleted and a window will popup displaying information as shown below. This will also automatically retrain and update the model and

encodings. After that, or if the decides not to delete this person and clicks “No”, they will be brought back to the main window.

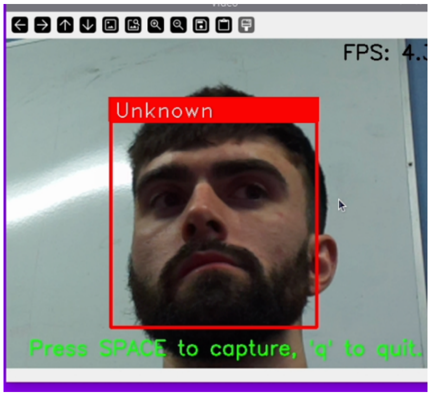


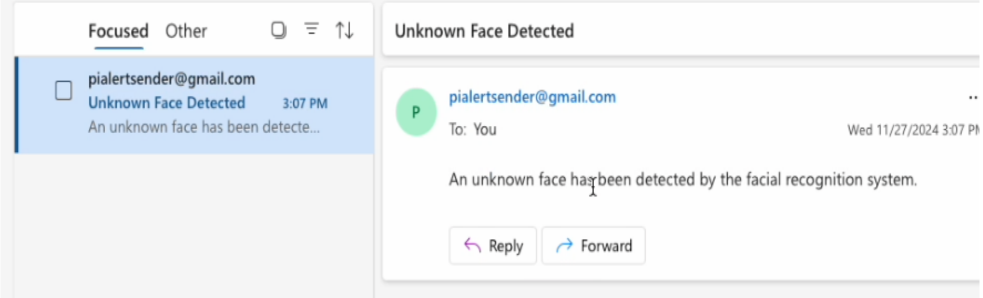
User 'Zane Smith' and associated photos have been deleted.

4. Selecting “Start Recognition” will start up the camera and begin facial recognition system. If a person is within the frame, and in the system, it will place a green box around them, along with their name. This will also remove them from the absent list.



An unknown person will showcase a red box around, triggering an email alert to be sent out. The example below demonstrates this happening after I had removed myself from the system.





pialertsender@gmail.com
Unknown Face Detected 3:07 PM
An unknown face has been detecte...

Unknown Face Detected

pialertsender@gmail.com
To: You
Wed 11/27/2024 3:07 PM

An unknown face has been detected by the facial recognition system.

Challenges and Solutions

1. **Challenge**: Hardware choices and issues.

   Solution: The first Raspberry Pi I ordered had hardware defects and had to be replaced. Waiting for the first Raspberry Pi to arrive, trying to troubleshoot and resolve the issues with it, and waiting for the replacement to arrive set me back an extra 2 weeks, in addition to joining the course late. Furthermore, once I got deep into testing and running the code, I was experiencing performance issues due to throttling. After picking up an active cooler for the Raspberry Pi, these issues were diminished. Also, the camera I chose for the project was not an official raspberry Pi camera and took a good chunk of time to figure out how to get the camera to initialize properly. Lastly, in a fully functional system I would upgrade to a higher quality camera, with a wider field of view, to improve results.

2. **Challenge**: Email alerts were being sent out in every frame that contained an unknown user. This meant multiple emails could be sent out per second or if a person is just beginning to come in the frame and isn't immediately recognized it would send out a false alert.

   Solution: The workaround for this issue was to implement a time-delay on the alerts. After the fix, if an unknown user was found it would email the alert, and then wouldn't send another until the 30-second cool down was over. Further changes were made to require an unknown user to be in multiple consecutive frames before the alert was sent out. To continue to address the issue I am working to store details from the unrecognized faces, allowing the system to distinguish between previously flagged and genuinely new unknown individuals, in the case of multiple unidentified users in a single frame, or multiple found within the cooldown time. Additionally, the email will contain a photo of any unrecognized faces.

3. **Challenge:** Adding users and taking photos for two people with the same name will lead to photos for both people be stored in the same place causing in inaccurate results.

   Solution: The first attempt to resolve this issue involved checking to see if the new person that shares the name matches the previous photos taken, and if not store separately. However, this results in no good way to differentiate between the two people in the system while live facial recognition is active. To solve this more data needed to be stored around the user. For example, it is possibly for multiple people in the system to share a name, but unlikely to share several combined characteristics such as first name, middle name, last name, birthdate, and position title.

4. **Challenge:** Scalability.

   Solution: The system was tested with a limited dataset (~15 users) due to the privacy and security concerns within a school setting. This many faces allowed for accurate results, however, much more testing is needed to see how accuracy and performance is impacted with many more users.

Degree Relevance & Final Outcomes

This project mirrors and integrates the knowledge and skills gained throughout my core Computer Science coursework and electives. It showcases how the theoretical concepts and practical techniques I have learned were applied to design and implement a real-world solution. Firstly, all computer science core courses (such as Computing 1, Computing 2, Data Structures, etc.) have given me a solid understanding of the fundamentals required to understand and begin any programing assignment. Concepts such as modular coding, debugging, code optimization, and algorithm design taught throughout programming courses, were used to implement features applied all throughout the project. The use of Python libraries (e.g., OpenCV, face-recognition, Tkinter) reflects the importance of leveraging existing tools for efficient development, a skill emphasized in coursework. Human Computer Interaction guided the decisions made during GUI development, ensuring the interface was accessible and user-friendly, allowing non-technical users to interact easily with the system. Operating systems provided insights into understanding the hardware and software interactions seen throughout the project. The Introduction to Internet of Things elective provided my first experience to work with IoT devices, and learn about their architectures, communication mechanisms, and operating systems. The Biometrics elective I took gave me a comprehensive understanding of computer vision concepts, which were essential for processing and analyzing visual data.

The project resulted in the successful creation of a reliable, affordable, and easy to use security solution that enhances security. The project met all outlined goals and did so at under $100. As mentioned in the challenges section, there are limitations, and extensive testing would need to be conducted if the system were to accommodate a drastic increase in users. On the other hand, misidentifications, with a limited number of users, were rare and were primarily limited to the initial frames as a user enters the cameras field of view. As is, the project serves as a budget-friendly security solution, which would be predominantly useful to provide an additional line of defense, particularly within small-scale environments like offices, rooms, or schools.

Conclusion

I have often found that the concepts covered within my Computer Science degree were difficult to grasp due to the heavy focus on theoretical background without adequate exposure to real-world applications. These projects have helped bridge the gap between academic knowledge and practical application. Being able to utilize the concepts and skills I've gained throughout my degree in a workplace setting has greatly enhanced my understanding and allowed for professional growth in areas that courses could not.

Overall, I successfully completed two degree-related projects that effectively showcase how my Computer Science degree has equipped me with the tools and knowledge to design and implement innovative solutions to real-world problems. The hands-on experience gained through these endeavors has been extremely beneficial in regard to preparing me for future technical challenges and career opportunities. This practical application has solidified my grasp of principles seen throughout the projects, providing a strong foundation for continued growth and success in my field.