Automated Team Attendance

FINAL REPORT

19
Mohamed Youseff Selim
Mohamed Youseff Selim
Lance Demers – Chief Engineer Hardware
Brandon Johnson – Team Leader
Nathan Oran – Documentation Manager
Angela Schauer – Meeting Facilitator and Chief Engineer Software
Connor Sullivan – Testing Lead
sddec20-19@iastate.edu
https://sddec20-19.sd.ece.iastate.edu

Revised: 11/15/20 Final

Executive Summary

Development Standards & Practices Used

This project was developed using an agile workflow. We utilized paired programming and code reviews to make sure our software is high quality. We developed a modular software system to simplify potential = future development and save ourselves time in case we needed to reimplement specific parts of the code.

Summary of Requirements

- Hardware
    - Project must run on a Raspberry Pi
    - Camera must be able to take clear pictures
    - Must be mounted in a way for the camera to see all of the classroom
- Software
    - Control the camera to take images of the classroom
    - Accurate detection of students from taken images
    - Determine which groups have absent members
    - Consolidate data and send a report to the professor
    - Professor must be able to create and adjust multiple seating charts for the system to work with

Applicable Courses from Iowa State University Curriculum

- CPR/S E 185/E E 285/COM S 227 - Introduction to programming
- COM S 228 - Data Structures
- COM S 309 - Application Development
- CPR E 288 - Embedded Systems

New Skills/Knowledge acquired that was not taught in courses

For several members of our team, the python programming language will be a new skill. This project also requires Machine Learning using Yolov3 and openCV which are new skills for all of us.

Table of Contents

List of figures/tables/symbols/definitions

Figure

Table

1 Introduction

1.1 Acknowledgement

We acknowledge Mohamed Youseff Selim for coming up with the idea of the project and also guiding our team through its design and development.

1.2 Problem and Project Statement

No doubt, Team-Based Learning (TBL) positively influenced attendance in the classroom. However, teams are still suffering from absenteeism. Although a handful of attendance tools exist, none of these tools are adapted to the team-based classes, besides, none of them records the attendance without any interaction from the instructor/student. Moreover, these tools are consuming about 2 to 3 minutes from the class time, which will sum up over the semester to 135 minutes in the worst case. The proposed tool will record attendance at zero time.

1.3 Operational Environment

The Automated Team Attendance Tool will operate in a classroom environment. The final product will be mounted in classrooms used by Instructors. The product should be able to function in small and large lecture halls.

1.4 Requirements

1.4.1 Engineering Constraints

- We were limited by the density which the detection software could work, so we developed a method to break the image into smaller sections to be able to accurately detect the objects we desired.
- Covid limited our ability to gather proper testing data, and properly test our project.
- With our design we are limited on the placement of the camera since it requires being plugged into an outlet for power.

1.4.2 Non-Functional Requirements

- The tool should be faster than previous attendance methods
- The tool should be just as accurate as previous attendance methods.

1.4.3 Functional Requirements

- Data Collection
  - Image collection of the classroom, over multiple instances during class
- Detect Students
  - Optimize the accuracy of student detection using deep learning analysis
  - Determine Which Groups Have Absent Members:
  - Compare the locations of students to a provided team-based seating chart
- Send Report to Professor
  - Consolidate the collected data into an email sent to the professor of the class
- Adjustable Seating Arrangements
  - Professor has access to create and update team seating charts

1.5 Intended Users and Uses

The product is intended to be used by any instructors or teaching assistants wishing to record attendance automatically.

1.6 Assumptions and Limitations

Assumptions

- Instructor or TA will provide a seating chart
- Teams will be in the same spot for the whole semester
- Instructor or TA will decide when to capture the attendance
- Product will be used in a classroom environment

Limitations

- Algorithm struggles to detect objects that are densely packed
- Test to see how program performs on models of the raspberry pi that have lower performance specs

1.7 Expected End Product and Deliverables

May 2020: Test product having minimum functionality. Such as determining the number of students present for each team. Being able to take a seating chart as input to be used to determine any missing students.

December 2020: A final product that will capture the number of students present for each team based on a seating chart provided by the instructor. Once the program finishes it will email the instructor the number of students present for each team.

2. Specifications and Analysis

2.1 Proposed Approach

The proposed approach to solve the problem of lengthy attendance times in TBL classrooms is using object detection to take attendance. A camera will take a picture of the classroom and verify it against a seating chart provided by the professor. An email with information about the attendance of each group will then be sent to the professor. As of now, we have been researching and testing various free, open source object detection software on pictures of lecture halls to see if they are a viable option. For this project, no specific standards need to be followed.

2.2 Design Analysis

After researching the available object detection software, we found that this proposed approach is feasible, specifically using the software YOLOv3. YOLO builds on top of OpenCV to create a system that can quickly and accurately detect and identify objects in a picture. To take the picture, we will have a camera connected to a RaspberryPi which will be connected to a server where it will run our code. We will be using Python for all the code relating to the object detection. For our project, we will take the data of the x,y coordinates of students that are detected and parse them so they are ready to be compared against the seating chart provided. To get the data of the seating chart, we will need a UI for the professor to use and analyze the data they give us.

2.3 Development Process

We will be using Agile processes. We want to be able to create code incrementally and include a lot of user input, so we decided Agile was best.

2.4 Conceptual Sketch

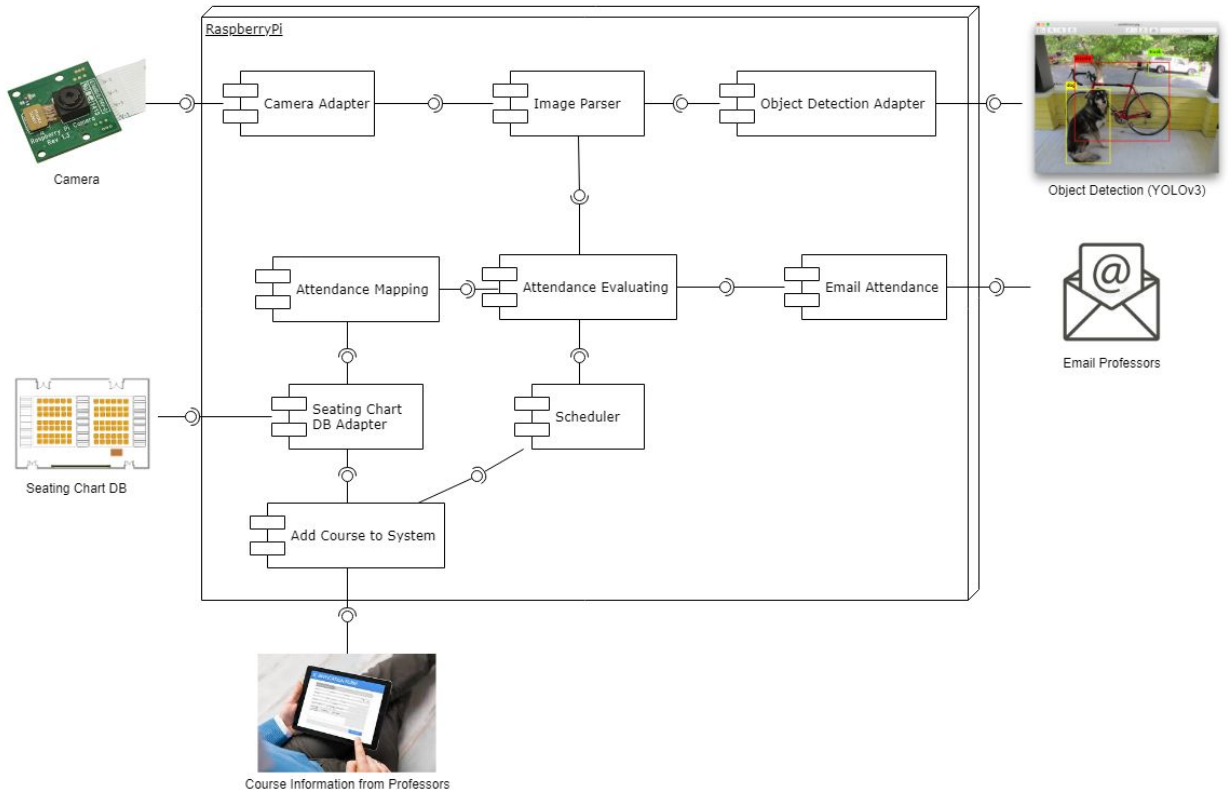

Figure 1

3. Statement of Work

3.1 Previous Work And Literature

There are similar tools available to professors for taking attendance in large lecture halls. One product that is mainstream and widely used is top hat. Top Hat is designed for in-class quizzes and tests for large lecture halls. Additionally, many professors use Top Hat for attendance because Top Hat shows the students who participated in the in-class quiz and who did not. The disadvantage of Top Hat is that it takes time to record the attendance in class and wastes valuable time. Our tool does not require any class time to take attendance and the time saved used to teach students. Another similar product available is called Spotter. Spotter is an app that students install on their phone and it compares a students location to where the lecture hall is to take attendance. Spotter does have advantages when compared as it requires no camera or hardware to work, the students just need to download the app. It also will be more accurate as it tracks the individual student and our tools just identifies group size groups. One disadvantage of spotter is privacy. For the spotter app to work, it needs to know your location. This requires you to put a lot of trust into the app that it will not track you at all times. Our tool does not require any individual information and will not even know your name. The biggest privacy concern of our tool is that the raspberry pi will have pictures of the classroom stored on it temporarily.

3.2 Technology Considerations

Our project uses a wide variety of different technologies, The technology is split between the hardware and software. The hardware we use needs to take clear pictures and needs enough computing power to run our Automated team attendance software. The tradeoff with the hardware is the cost versus the computing power. Our goal is to minimize cost while being able to run the software effectively. The choice for our software technology is less straightforward. Our primary concern with our attendance tool is accuracy. We do not want to mark students as absent if they are present. Additional considerations are what programming language, libraries, and what machine learning framework / algorithm to use. For this project we chose python of other languages because it allows for a faster development process. The libraries that we are using are OpenCv and YoloV3. We chose these libraries because of the amount of documentation available and the reputation they have in the computer vision community. YoloV3 is also known to run much faster than other object detection algorithms and is important because a raspberry pi has limited resources.

3.3 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and to understand interdependence among tasks.

Our group split the responsibilities of the project into two main groups, software and hardware. The software team will focus on the algorithm of the automated attendance tool itself. The hardware team does not only set up the hardware, they are responsible for setting up the platform itself. This includes the backend which sets up the raspberry pi as a server that schedules the tasks to run at the appropriate time. One of the largest tasks for the hardware team is setting up the email server. The email server will email the instructors the attendance results for the day.

Both the hardware and software team will decompose their responsibilities into smaller tasks. The software team will need to complete the following tasks to insure that the object detection algorithm works. First they need to make the object detection as accurate as possible. This includes combining opencv, yolov3 and so given an input image, it can detect all persons and store their coordinates. Another task they will work on is determining what people belong to what group. Finally the last big task is comparing the group attendance we have to the seating chart given by the instructor. In addition to setting up the email server, the hardware team also needs to set up the hardware and scheduling. The hardware requires creating a program that sets up the camera on the raspberry pi and taking and saving pictures. The scheduling tasks include, running the attendance software multiple times and then sending the results to the instructor. The pictures taken will also need to be deleted for privacy reasons.

3.4 Possible Risks And Risk Management

We will need to make sure the program detects important data as accurately as possible, and is able to compare that data to a given seating chart. We will manage this by taking multiple photos if needed to make sure that the data we receive can produce the desired output.

We will need to make sure to consider the privacy of the students, and anyone who ends up being in the picture as well. We will manage this by deleting the image after the image has been successfully used to determine attendance, and after a predetermined time length. Any pictures that are either invalid or are determined to be bad will just be discarded immediately.

Since our raspberry pi will be using raspbian as our operating system, we can implement this through the use of cron jobs. Cron is a CLI task scheduling program. In addition to deleting the photos, cron will also schedule all the tasks. The timeline of all the tasks executed is the following:

1. Take photo of classroom
2. Run the object detection program.
3. Convert results to excel file.
4. Email instructor the results.
5. Delete photo.

3.5 Project Proposed Milestones and Evaluation Criteria

Our first milestone will be being able to further develop the algorithm to be able to detect larger numbers of students grouped closely together. We can test this by taking some stock photos of large groups of people that we can determine the number of people in the photo, and running the algorithm on the photo to make sure it can count that many people.

Our second milestone will be being able to take a photo from the raspberry pi and comparing it to a given seating chart. We will test this by providing simple images to use that contain a small group, or 2 small groups, and then use a seating chart that we know what the result of the attendance should be.

Our third milestone will be being able to run the scheduled tasks. This includes being able to run the attendance software, possibly multiple times, emailing the professor the reported attendance, and deleting the picture after the program is finished with it. We can test this setting some output to see that our program can be automatically executed multiple times based on some given criteria. We can set up the raspberry pi to email one of us the results, or all of us in the case that a professor would like multiple people

to be notified of the attendance. We can check to make sure that once the program is finished it no longer contains the image in memory.

The table below is our timeline for our big picture milestones. We will be going more in depth into each of these dates and milestones in Section 4.1.

09/29/2020	Milestone 1: Student detection and attendance mapping
10/06/2020	Milestone 2: Ability to capture and process images on the raspberry pi
11/03/2020	Milestone 3: Ability to submit seating charts and schedule attendance capture

Table 1

3.6 Project Tracking Procedures

We will be using a Trello board to keep track of our progress on the project. We will be using the typical backlog, doing, and done trello tabs to keep track of our work. What this means is that when we begin working on any component of our project, the team will generate all the tasks that need to be completed for that component. Each of those tasks will be represented on our trello board and start in the "Backlog" list. From there, we will evaluate each task on its priority and effort. Priority is determined by dependency for other tasks and importance to the big picture of the project. Effort is determined by the amount of time both in task completion and research necessary to complete the task. We will create a labeling system to denote the priority and the effort and mark each card accordingly. From there, we will then arrange the tasks based on priority. High Priority tasks will be at the top, and low priority tasks will be toward the bottom. Then, starting from the top of the list, each team member will choose a number of tasks that they think they can complete before the next time we meet and put their name on those cards. When anyone begins working on one of their cars, they will move it from the "Backlog" to the "In Progress" list. This way the rest of the team can see what everyone is doing at any time by checking the trello board. Once someone finishes a task, they will move the card from "In Progress" to "Done."

3.7 Expected Results and Validation

We will determine the success of our solution by the solution's ability to consistently deliver the attendance information of each team in a given team-based learning class, by the number of persons each team has present (rather than individual attendance). In particular, this solution as a whole will be considered a success if it is less time consuming than manually taking attendance.

We will determine the success of our data collection requirement by our solution's ability to regularly collect images of the classroom at the correct scheduled times, proper locations, and done the proper number of times.

The success of our student detection requirement will be validated by determining whether the solution can use the collected images to correctly identify where all the students are in the given classroom, map each student to their correct team, supplied by an attendance sheet, and determine which teams are then missing students.

The success of our requirement to send the report to the professor will be measured by whether the system can reliably send the team attendance to the professor at the correct times.

Finally, our requirement for adjustable seating arrangements will be validated by our solution's ability to receive updated attendance lists and sheets from the professor, and apply these changes to the seating chart and/or schedule prior to the next class meeting time.

4. Project Timeline, Resources, and Challenges

4.1 Project Timeline

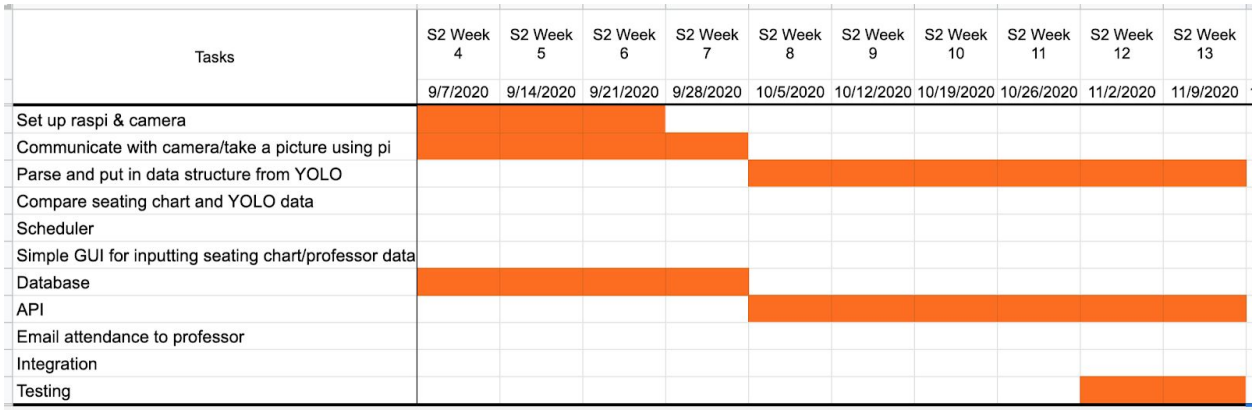

Figure 2

Dates of Deliverables

09/28/2020	Raspberry Pi fully configured
10/5/2020	Communicate with camera/take a picture using pi
10/5/2020	Database tables constructed
11/16/2020	API Completed
11/16/2020	Testing Yolo on pi complete

Table 2

4.2 Challenges

We thought the full completion of this project was possible within the two semesters given. However, this project requires a wide breadth of development that needed consistent progress throughout the time we had. Specifically, the greatest technical challenges were learning to use and train the Yolov3 Neural Network with OpenCV to accurately find students and retrieve the locations of students on the image.

Additionally to our major technical challenges, we needed to develop an accurate seating chart mapping algorithm, configure the system to email results to professors, create an entire web application for professors to submit seating charts and schedule when the system takes attendance, configure the raspberry pi to interact automatically with these elements, and, finally, construct the permanent mounting system that can work with any of the classrooms that use team based learning. Initially, we didn't see any of these tasks as too complex for us to handle, but there are basically five or six large tasks that would take the efforts of all of the members of the team to complete.

Unfortunately, due to the coronavirus pandemic, development of this project became difficult. The shift to working entirely virtually made keeping in sync with each other an obstacle. Additionally, the limited access to Iowa State classrooms and reduction of in-person Team-Based Learning classes this semester made a true implementation of our project impossible. Also, the majority of the team never had access to the raspberry pi, which limited the ways that we could split work and effectively complete tasks. All these issues together severely hindered our ability to keep up with the workload we set for ourselves and resulted in an incomplete project.

4.3 Personnel Effort Requirements

The following table breaks down the amount of time for each member of the team spent on the project over the course of the two semesters. Lance, Angela, and Nathan ended up focusing on the Database and API, and Brandon and Connor worked with the pi and YOLO.

Student	Hours
Lance Demers	56
Angela Schauer	67
Brandon Johnson	51
Connor Sullivan	58
Nathan Oran	68

Table 3

4.4 Other Resource Requirements

To properly create our project, we needed test data. This test data would ideally be pictures of people sitting in a setting close to one of the Team Based Learning classrooms at Iowa State. We collected this data on two occasions, however due to the limitations of the coronavirus pandemic, these test images did not fully meet the needs we wished for this project. Our team was able to capture images of ourselves in a Carver lecture and we were able to recruit a few test subjects of a small off-site room to do some initial YOLO testing as well.

4.5 Financial Requirements

For this project, we needed to pay for: RaspberryPis and cameras. Below is the

ITEM	COST
Raspberry Pi Model 4	$35
Micro SD card	$5
Micro USB Charger	$5
Pi Camera V2	$30
Mount/Case	~$20

Figure 3

5. Testing and Implementation

5.1 Interface Specifications

We are not currently working on any hardware/software interfacing for testing our project. We do have communication between our software and a RaspberryPi/RPi camera, so some form of testing will have to be created in the future, but we have not worked on this part of the project yet.

5.2 Hardware and software

In terms of Hardware, we only used the raspberry pi and the raspberry pi camera for our project. The rest of the work was development on the raspberry pi and software work. When testing the system in a lecture hall setting we mounted the raspberry pi and camera in various locations throughout the front of the class room in order to have test samples from various angles and reference points. The use of the Facade pattern helped to streamline testing that any third party hardware, i.e. the Camera, was correctly integrated and configured.

On the front end of the software, we developed using Python for the various components that run locally within the raspberry pi. On the raspberry pi, we wrote adapters for the camera and the object detection software, and functions that would use those adapters to take a picture and output the locations of the students identified in said picture. Much of the logic within the raspberry pi was centered around object detection with YOLO. We discovered a few obstacles with YOLO during our development: notably that it would often clump densely clustered groups of small objects together into a single object. To curb this we developed code that would partition a given picture into several smaller pictures that would then be individually processed by YOLO. Additionally, YOLO naturally detects a fairly wide set of different objects other than just people, or detects the same object several times. We adjusted the YOLO implementation to account for both of these issues.

Beyond YOLO, one fundamental aspect of development on the raspberry pi was determining how to pass the images between the various components. Given that passing images as parameters in Python or through the shell is a rather cumbersome process, we decided to simply pass the image file's name instead for the relevant functions. To help support this protocol, we created directories for initial camera captures files as well and another one to store the processed results from YOLO. While the processed YOLO images are not inherently required for this system to run, they do serve as an invaluable tool for confirming that YOLO is correctly detecting the students.

For the backend, we used MySQLWorkbench to create and host our database on our virtual machine provided by the ETG. Our database consisted of five tables: Professor, Classroom, Class, SeatingChart, and Times. The Professor Table stored data for each user of the application. Their email address was used as the primary key for this table. The Classroom table stored the building and room number for the classroom as a single concatenated string to function as the primary key for the table. The Class table stored data related to specific classes, the course name and section, a foreign key to the professor of the course, a foreign key to the classroom which the course would be held in, and an auto generated id for the class. The SeatingChart table was designed to hold information about each student's seat in every class. It had an auto generated id, a foreign key to which class it was in, a foreign key to which classroom it was in, a seat position within the classroom, and the team name label for that student. Finally, the Times table held data for all the times that a professor requested an attendance capture to be taken. This table has an auto-generated id, the time at which they'd like to be captured, the days of the week that that class is held, and a foreign key to the class

itself. The idea behind this table is that all that the raspberry pi cares about is when to take the picture and who to send the email to. So, if the pi could collect all of the times for its classroom on each day of the week, it could create a schedule from a call to this table alone.

As for the API, we decided to implement an ASP.NET API with entity framework core. We established the connection to our database using a DBContext object. For each table, we have a DBSet, Controller, and Interface to implement a full set of CRUD operations as well as a few more that allow for more streamlined API calls from the eventual User Interface and calls from the raspberry pi. These interfaces are implemented via services that are scoped on startup to perform some necessary dependency injection for the services to work. Our architecture is boiled down to the following diagram for each of the tables in the database.

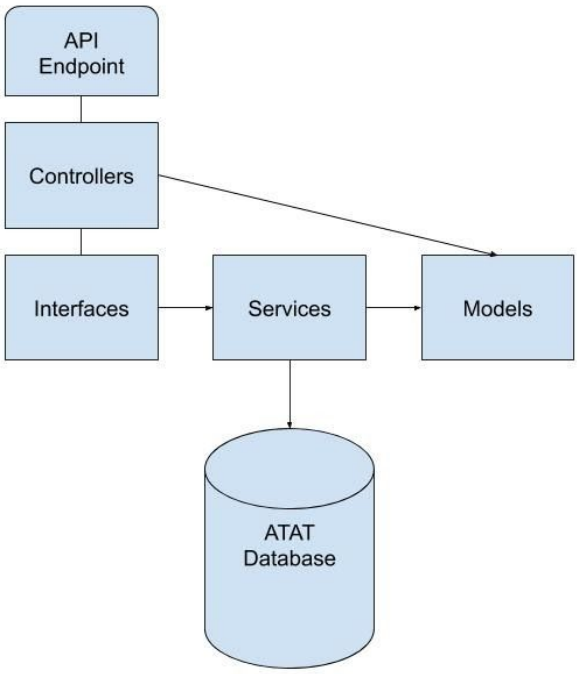

Figure 4

5.3 Functional Testing

- We will be using test images to verify output from our program.
- After implementing the scheduler we will check to make sure each task executes, and in the correct order.
- When we get the email server up we will send out test emails to ourselves.

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

- After successfully implementing our project on our Raspberry Pi, we will test the effectiveness of our system on a less powerful, but less expensive version of the Raspberry Pi to test for more cost effective alternatives.

- Should our solution prove effective in the supplied tbl-classroom layouts we will test its effectiveness in different classroom layouts.
- After implementing the image deletion in the scheduler, we will make sure the images are successfully deleted from the Pi after use.

5.5 Process

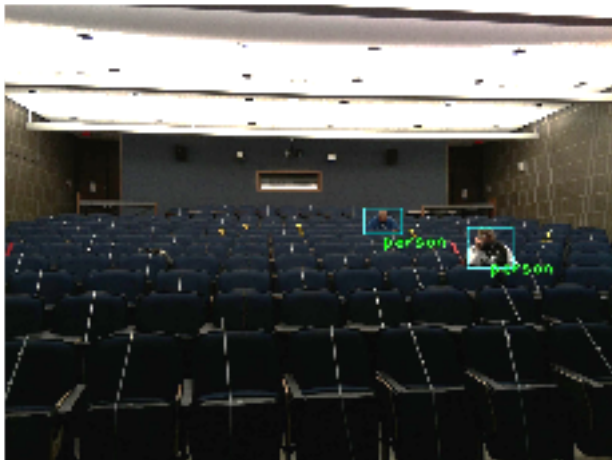

Figure 5

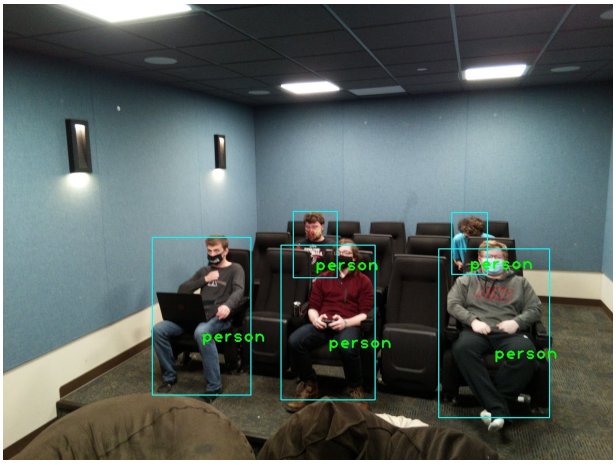

Figure 6

Due to the restrictions of COVID-19, and in-person basis of this project, it was difficult to test our system. During Spring semester, we used images of lecture halls from the internet to do some initial testing with the object detection software. We were able to meet once during Fall semester to set up the Pi and take an image of a classroom at Iowa State with our senior design team in the seats. An additional test was done with a few more subjects at an off campus site. These tests consisted of capturing images of students in seats and running them through our YOLO process on the raspberry pi.

The team working on the backend used MySQLWorkbench and Postman to test the API and Database.

5.6 Results

Because Yolov3 is processing visual images, the first ways that can verify the results of our tests are by looking at the output image from Yolov3, and determining if it successfully and accurately detected the students in the classroom. It is possible to eventually develop automated tests using image comparison once we have images for the expected output, but the first version of any visual automated test, or any test is manual.

What we have found so far in testing Yolov3, is that it struggles when there are a large number of students, close together in the test image. After a certain point, the algorithm just puts one large box around the back of the class and calls it a person. One solution to this accuracy issue is that we can split the image into multiple smaller pieces and have Yolov3 process each piece separately. We've found that apart from some other double counting of students on the edges, this helps with our accuracy issue.

Currently, we haven't done any modeling or simulations for the mount of the pi and the camera. Possible future modeling could include 3D printed camera mount. For our software, we need to as a group come together and solidify better testing processes and look into more efficient ways of testing within Python and Cron.

6. Closing Material

6.1 Conclusion

For the first semester of our senior design project we achieved several key technical tasks that help us achieve our goals of creating an automated team attendance tool. The first major task completed is being able to detect people. This was done by using the neural network YoloV3. The other major task completed was the design for the project. Since the design is done, we just need to start the implementation. For the second semester of senior design we achieved several other key tasks. On the hardware side we implemented an algorithm to split up the image to increase accuracy of detection, created an adapter to use the camera attachment, and developed a main method to call and run all of the methods necessary for the pi to operate. On the software side we developed the database and API for both the database and website GUI for the seating chart and course submission from the instructors. There is still plenty of work to be done on this project. We still need to implement the seating chart mapping, the scheduler for the pi, the program to send emails to professors, and finish the GUI for the website.

6.2 References

Redmon, Joseph. "You Only Look Once: Unified, Real-Time Object Detection." *Pjreddie.com*, University of Washington, pjreddie.com/media/files/papers/yolo_1.pdf.

6.3 Appendices

6.4 Appendix I - Operational Manual

No instructional steps to report currently.

6.5 Appendix II - Alternative Versions

https://docs.google.com/document/d/1ply5mAkCPSNQ9d2LkwEtzEm6ymEwCBv-63Cyp3Zp06g/edit?usp=sharing

We decided to scrap this version, because it didn't include anything done in the second semester.

6.6 Appendix III - Other Considerations

None to consider here.