EE2 PROJECT GROUP 3 – AUTOMATED PEOPLE COUNTER



Final report – March 13th



Supervisor: Dr Tom Clarke

Name	CID
Omar Wolfe Hussein	01046652
George Padley	01088571
Jacob Kay	01052280
Zheng Lee	01042500
Laura Tuckey	01049760
Karmanya Sareen	01067937
Alex Karet	01078146
Samuel Yang	01067937

Imperial College

Contents Page

Section	Title	Page
1	Abstract	3
2	Introduction/Background	4
3	Design Criteria	4
4	Concept Development	4
4.1	Lasers	5
4.2	Computer Vision	11
4.3	Server	13
4.4	Hardware implementation	17
4.5	Connecting the systems for implementation	18
5	Meeting design Criteria	20
5.1	Social Impact	20
5.2	Design Criteria Check	21
5.3	Cost of System	22
6	Project management	22
7	Future work	23
8	Conclusion	23
9	References	24
10	Appendix	25

Table	Page Number
Table 1 : Pros and cons of Asynchronous and Synchronous FSMs	8
Table 2 : Truth table for multiple sensors	9
Table 3 : Table of database tables	13
Table 4 : Design Criteria checklist	21



Figure	Page Number
Figure 1 : System overview schematic	5
Figure 2 : Laser grid set up on the aluminium profile	6
Figure 3 : Schematic and breadboard design of the phototransistor detector system	7
Figure 4 : Oscilloscope trace of someone entering the room	7
Figure 5 : FSM Diagram for implementing the laser 'trip beams' synchronously	9
Figure 6 : Image captured from a video and image after background reduction	11
Figure 7: One frame which has the line and boxes for any detected contours and another frame with reduced background after dilation and erosion.	12
Figure 8 : Diagram layout on how centroids determine direction	12
Figure 9 : Concept art for the user interface	17
Figure 10 : Aluminium frame cross section - http://uk.rs-online.com/web/p/tubing- struts/7613284/	18

Appendix	Page Number
A) Meeting Minutes	25
B) Project Plan	27
C) Computer vision code	29
D) Laser Safety information	31
E) Complete code for Project	32
F) Market research - Email with fire service	38
G) Database Protocols	40
H) Design Specification	41
I) Cost Breakdown	45



1. Abstract

The thought behind this project was to assist Emergency Responders in the case of an emergency for a large building where there are many rooms containing a large number of people, so that once people are leaving the building the Emergency Responders are able to know exactly where people are stranded within the building. The system is made up of two detection methods Laser 'trip beams' and Computer Vision (CV) techniques. The data from each of these is sent to a database on a server and a web interface provides easy access to for the user. The system works successfully detecting multiple people accurately using both the CV system and the Lasers in parallel, taking readings and submitting to the server every 60 seconds. Also testing both systems with varying movements such as someone entering before turning back resulting in no change in the count for both systems, but also how the system works when multiple people enter in close proximity to one another.



2. Introduction/Background

This project is a system to determine how people are distributed throughout a large office building and how many people are in each room of the building. In the event of fire or other emergency the data can be easily accessed remotely by security, the building manager, emergency responders etc. and be acted on. The goal is that lives are saved as rescuers get an idea of where people may be trapped quickly on arrival.

As of the interim report submitted on 8th Feb 2017, the design specification had been written and a concept design was being developed. The concept is detailed in section 4. The work that has occurred since revolves around the technical implementation of this developed design.

3. Design Criteria

Since the submission of the interim report, the design criteria for the project has not changed. This is in Appendix H.

4. Concept Development

The concept chosen and developed is as described in the interim report. The system is made up of four subsystems:

- 1. The laser based detection system
- 2. The CV detection system
- 3. The processing and data communication system
- 4. The data logging and access system

The premise of operation is that the laser and CV detection method work independently to calculate the number of people in the room and the data is sent to the server that logs the data, processes it, and facilitates access to the data. The system overview can been in the diagram below.



Figure 1: System overview schematic

Two different detection methods are used as they provide redundancy for each other. The CV, whilst most likely more accurate than the laser based system, will fail quicker during a fire. This is because as a room fills with smoke, the camera - which operates with visible light frequencies - will be unable to detect any motion through the smoke. The lasers however, will easily penetrate the smoke particles and a sufficient amount of light will pass through to the detectors. This will allow for motion detection to continue even during a fire.

In addition, the laser system is cheaper to implement and will work very effectively on its own for narrow doorways where only one person can pass through at once. Therefore a potential solution for narrow doors is to just have the laser system and reserve the CV system for wider doorways with a higher throughput of people. This reduces the total cost of the solution to the client whilst maintaining fully operative and reliable devices.

4.1. Lasers

The lasers implement the basic method of counting people, as described in the proposal. It is designed to be cheap, accurate, and require no interaction from the user for it to work. The basic principle is using two lasers on opposite sides of the door (inner side and outer side) with phototransistors on the other side of the door to detect the laser beam. This is shown in the picture below.





Figure 2: Laser grid set up on the aluminium profile

The laser arrangement detects the direction in which the person is going, and whether they make it through the door completely and in/out of the room. There are two sets of these lasers placed at different heights in order to reduce the number of false detections (i.e. swinging arms excessively while walking through); but also if of one of the lasers becomes blocked somehow, meaning that the second laser on its side would still be working to detect someone entering or leaving. Both sets of lasers work together as one system.

The laser system must accurately detect a person going through the door, determine the direction of the person and update the internal count of the room appropriately. It also has to deal with

4.1.1 Implementation

The laser system is implemented using 5 mW red (650 nm) lasers, SFH300-¾ phototransistors as detectors, and the Python based '*pigpio*' module (1). The circuit was designed such that when the beam is unbroken, the phototransistor is saturated and produces an output of approximately 3.3 V and when the beam is broken, the output voltage falls to approximately 0 V. This was done through a simple potential divider that biases the phototransistor as well as limiting the output going to the Raspberry Pi to 3.3V which is its operating voltage. The phototransistor is used in the common collector configuration.





Figure 3: Schematic and breadboard design of the phototransistor detector system

The Raspberry Pi does not come with a built in Analogue to Digital Converter (ADC), so can only read the signal on its GPIO pins as a digital signal. The Raspberry Pi operates at 3.3V logic, an input lower than 0.8 V is read as LOW, and an input above 1.3 V is read as HIGH (2). Since these are typical values, it must be ensured that the voltages going to the Raspberry Pi are never near these thresholds so that there are no errors reading the signal from the phototransistor. This is why the detection circuit is designed to give outputs of $\simeq 0$ V and $\simeq 3.3$ V ensuring there is never the possibility that the Raspberry Pi reads the signal incorrectly. Figure 4 below shows the oscilloscope trace from a pair of sensors. The sharp edge is exactly what was desired as it doesn't give any ambiguity to the digital inputs of the Raspberry Pi. A person walked through the door to give this trace. It shows how one laser beam (green trace) is broken before the second beam is broken, and they are unbroken in the opposite way.



Agilent Technologies

Figure 4: Oscilloscope trace of someone entering the room



Initially the system relied on interrupts that detected the rising edge from each sensor (i.e. when the laser beam was unbroken) and setting a timestamp for each laser being unbroken. This worked well until a person would walk in through breaking both lasers, then decide to walk out again hence unbreaking the laser beams in the incorrect order. This caused the system to count the person going out of the room when in reality they went in and then out. There were other issues with the implementation of a timeout in the case of only one laser beam being broken. The drawbacks of using this initial approach became visible and a new approach was necessary in order to make the system robust to all situations.

Using a Finite State Machine (FSM) solved all the issues presented previously. An FSM allows all the possible inputs to be accounted for. Only the correct inputs to the FSM in the correct order would cause it to count a person going in/out of a room. Any other possibilities (as described above) will not trigger a count increase/decrease. It is an inherently more accurate, robust, and stable solution. However there was still a choice between using an asynchronous FSM, using interrupts, or a synchronous FSM using sampling. The main advantages and disadvantages of both options are listed below in table 1.

Sampling	Advantages	Disadvantages
Asynchronous	 Won't miss an input Only uses CPU when a rising/falling edge occurs 	 State diagram more complicated Harder to figure out the state transitions
Synchronous	Simple state diagramMore resilient to glitches	 Might miss an input if sampling rate is too low. Using CPU to sample constantly

 Table 1 : Pros and cons of Asynchronous and Synchronous FSMs

The FSM was implemented in Python, using an open source FSM code (3). However the code used was written in 2002, based on Python 2 and since the codebase used for the project is based on Python 3, the FSM code was mostly rewritten. Both FSMs were implemented for testing to decide which option would be the most reliable. From the testing, the asynchronous interrupt based FSM performed worse with more glitches and undefined state transitions occurring. The synchronous FSM that sampled the inputs every 1 ms also had some undefined state transitions, but it was easier to figure out what caused and add appropriate state transitions and adjust the sampling rate. The inputs to the system are read using the *Pigpio* module (1).

Using an FSM allows easy combination of multiple heights of lasers into one system. Currently the implementation is simple. Both lasers on one side of the door (the 'inner' side or 'outer' side) are paired together through logic. This is summarised in the truth table below. The input from the sensors is defined as: a 0 means laser beam is broken, and a 1 means laser beam is unbroken. This is why the logic to combine the sensors inverts this, so that the FSM makes more logical sense i.e. a 1 will mean that side

Input from one side's sensors	Inversion	Output	Meaning
00	11	1	Both beams broken
01	10	1	One beam broken
10	01	1	One beam broken
11	00	0	No beams broken

is broken and there's a person there. The implementation is a simple NAND gate done in software, to give the input to the FSM.

Table 2: Truth table for multiple sensors

For n number of lasers on one side of the system it can be simplified by to : $S = !(S_1 \& S_2 \& ... \& S_n)$.

The inputs from the 'inner' side and 'outer' side of the door are concatenated into one string. Which then is processed as the input to the FSM. The scope trace in Figure 4 provides context for the FSM, bearing in mind the output shown in Figure 4 is inverted in software to give input to the FSM.



Figure 5: FSM Diagram for implementing the laser 'trip beams' synchronously

From the state diagram (Figure 5) it can be seen that only the specific sequence of laser beam breaks/unbreaks will count as a person going through the door. This is irrespective of how long this takes (i.e. if a person stops in the door). From the inversion of Figure 4 the FSM can be followed easily.

An issue that arose with this implementation was the scenario in which one of the sensors is blocked, hence giving a false positive. This issue was resolved by the timing the duration for which the false positive was received, for example 5 seconds. Hence, if the sensor gave a positive for a time longer than the threshold time, it would be declared a false positive and the date from the sensor would be ignored when calculating the overall input from that side of the doorway.

Testing of the laser system proved highly successful. In the case of people walking through being unaware of the systems method of counting people, the accuracy hit upwards of 90%. When people became aware of the way the lasers work, accuracy dropped to approximately 80%. This was because the testers were walking through with the intent of causing false detections, for example swinging their arms excessively or walking through purposely holding something in front of them. Since the final implementation will use non-visible infrared transceivers, the accuracy should be above 90% as people would treat it as a normal door.

The full laser code is included in appendix E as part of the overall system code.

4.1.2 Laser Safety

The lasers being used in the project emit a laser beam of \simeq 5 mW which is acceptable to be used without extensive safety requirements. However, the department required following safety guidelines as set out below to ensure that no harm could come to people around the project. The lasers were deemed to fall into the laser category 3R - as their output power is around 5 mW - which means that the following steps must be carried out, for the full safety requirements please see appendix D:

- Do not stare into beam
- Avoid setting up beam at eye level
- Terminate all beams with opaque stops within limits of experimental area
- Direct beam away from populated areas and entrance doors where possible
- Avoid accidental reflections from watches, belt buckles, jewellery and other equipment
- Ensure laser is mechanically secure, and electrically tested to College standards
- Ensure laser displays correct labels according to latest BS EN 60825 standard, particularly blackon-yellow triangular "laser starburst" sign

In order to comply with these requirements, added safety measures were implemented to the project. Black card was positioned behind the lasers and the detectors to ensure that the beams won't pass outside of the bounds of the detectors and is contained. In addition, "laser starburst" signs are displayed facing all directions on the door frame to ensure that everybody around the system is aware that lasers are being used.

To satisfy the outstanding requirements, operating procedures were developed. These are as follows:

- The lasers will not be left unsupervised while powered on and emitting light.
- Any people walking through the door will cover or remove belt buckles, shiny jewellery etc.
- No persons will be permitted to look into the lasers or their immediate reflections

On 13/3/17 Andrew Holmes inspected the system and deemed it safe for use in the labs and demonstration.



4.2. Computer Vision

As mentioned in the interim report. The software required in order to enable the detection of people is as follows

- 1. Capture images from the camera frame by frame
- 2. Pass the frames through object recognition algorithms, from OpenCV
- 3. Track the objects' movements across the field of view
- 4. Determine direction of motion of the object
- 5. Update net movements count based on direction of movement

The program uses the camera to capture an image frame by frame. Then it detects moving individuals so by using background reduction. Background reduction is a technique used to generate/extract a foreground mask of a moving object in a static background (4). Once background reduction is used, any moving object detected will produce a white blob and the rest of the background will be black.



Figure 6: Image captured from a video and image after background reduction

The functions medianblur(5), dilate, and erode(6) are used to smoothen the blobs. Now that the program can detect moving objects, an algorithm is implemented to track the movements of the object/individual. Firstly, contours (curves surrounding the boundaries of an object) of the moving object must be located. Since contours are accurate and work effectively when using binary images, they are ideal for the images created by the algorithm that are black and white content only (binary). Once the contours are found, a rectangular box is drawn around the contour for easy detection of the counters.



Figure 7: One frame which has the line and boxes for any detected contours and another frame with reduced background after dilation and erosion.

In order to detect and count the number of people going in / out of a room or building, an imaginary line is drawn on the image which acts as the separation across the door. Therefore, a coordinate/point somewhere inside the contour must be fixed. However, a single centroid won't be able to determine the direction in which the object is travelling. Hence, two centroids are take none from the previous frame and one from the current frame. This enables the detection of the object's direction of movement by using the two centroids and the line.



Figure 8: Diagram layout on how centroids determine direction

The program is capable of detecting a maximum of two individuals crossing the detection line from one direction. This is done by calculating the area of the rectangular box bounding the contour, and then comparing it with a suitable threshold value. The program can be modified to detect more than two individuals by including more possible cases and converting the area into corresponding number of people.

When two individuals cross the detection line simultaneously from different directions, an occlusion occurs. Note: Occlusion is a phenomena where two tracked objects meet and form one big object. The program is still unable to handle occlusion and will usually result in an incorrect value. However,

modifications can be made by implementing an advanced algorithms (7) to handle occlusion and hence increase the reliability of the program. The python code for the CV is included in appendix C.

4.3. Server

The server is the central brain of the system. All the data is collated here before being backed up to locations off site. The server is also the location accessed by the emergency services in order to see the actual count from the building. The solution uses Ubuntu as the basis for the servers running an apache server on this operating system. In implementation, any Linux, Apache, MySQL and PHP based server would be suitable.

4.3.1. Database

The main purpose of the server is to act as the database for the system. This is where all the data collected by the sensors is stored and collated to provide useful information to the end consumers. The server receives data from many sources (of which the bulk just states a change in number), a sensor ID and a room ID, inputted to a single table. During normal use, the database will periodically perform lookups on this data to make adjustments to the table that stores the values of the number of people present in a room. The table storing the room values is the only one in normal use that shall be accessed by the user through a web page.

This is implemented on a MySQL server, using phpmyadmin as an open source software package to help development and management of the system. There are currently 7 tables in the system. These are shown in the table below.

Table Name	Function
PeopleCountLaser	Stores sensor data from the lasers. Has columns: PeopleCountLaserID SensorID RoomID Time CountChange
PeoplCountLaserHistory	Stores the history of the data from the lasers so further analysis can be applied. Has columns: PeopleCountLaserHistoryID SensorID RoomID Time CountChange



PeopleCountCV	Stores sensor data from the CV. Has columns: • SensorID • PeopleCountCVID • RoomID • Time • CountChange
PeopleCountCVHistory	Stores the history of the data from the CV so further analysis can be applied. Has columns: PeopleCountCVHistoryrid SensorID RoomID Time CountChange
Rooms	 Stores the count for each room. Has columns: Roomsid Roomid RoomName Total
Sensors	 Stores each sensor along with the rooms it is connected too. Has columns: SensorsID SensorID RoomEnterID RoomExitID
SensorHealth	 Stores data about the sensors for seeing how well they work. Has columns: SensorHealthid SensorsID OnSince MissedSends LastCheck

Table 3: Table of database tables

The system works by having the main PeopleCount tables updated with the changes from the sensors. On the database events are then run every 30 seconds, this event deletes all the data from these tables. The delete then triggers two more commands, these both occur before the delete will happen. The first copies the data to the respective history table, keeping it for later dates. The second performs a lookup on Sensors and then applies the change to the room's table total. Before inverting this number and applying it to the RoomExit in the rooms table. Currently, the system only applies the laser code to the room's total. In future, after more testing of the reliability of the system more analysis will be applied in situations where there is more than one sensor on a particular door. This can be read in more detail down below.



In order to increase the accuracy and reliability of the system, some simple rules are in use for the database. The first is that the count for no room shall be allowed to go negative. If a room change does cause a room count to go negative, the count of the room being entered will still increase while the room being exited will not be allowed to go below 0. Furthermore, certain path finding algorithms might be implemented in the future. Since certain rooms will be connected, these could then be decremented appropriately if it can be deduced that the person is likely to have originally come from a specific room and just hadn't been sensed leaving. This could be used effectively in the case of a fire or any other emergency where the software can assume that all people are attempting to leave a building. The system will be reset each night to have zero people in the building. This will be corrected if there are people in the building as soon as they pass through a door but will help reduce the chance of a room count just increasing if the sensors never sense a person leaving. This makes for a very robust system that should never deviate far from the actual value due to adjustments if a room goes negative and having a protocol in place to guarantee that the count does not only increase.

Since certain doors will have multiple sensors there will be the need to perform statistical analysis on the data that is sent. If both sensors reflect the same change, it can be assumed that both are correct. Problems arise when the database has to deal with sensors providing different data. The simplest way of dealing with this is to find the expected value of the data provided by the sensors using the accuracy found in testing. This would give reasonable results and using the systems described above the accuracy and reliability can be improved extensively. However, since it is easier for the system to correct a lower count rather than a higher count, it would be desirable to make sure the counts are always tending to the lower value. With more data available, more complex algorithms could be written which look at past data to determine the most likely result. For example if there is always a large influx of people at around 9am and one of the sensors is reading 0 and the other some value similar to previous values for this time period. Instead of averaging the values or even being cautious and going with the lower value it would instead be best to assume that the sensor reading 0 is not working and rely on the other sensor. This sort of analysis is the end goal of the server however, until a couple of systems are installed and real data has been collated and stored, it is hard to write any form of program to perform this as it is unclear what the data will look like in normal use.

Since the plan is to have the database performing large scale statistical analysis on the data it is also sensible to record as much data as possible from the sensors. Some examples of this data is sensor uptime, connection uptime, sensor errors (found by room counts going negative - although needs to be considered in nightly reset) and sensor conflicts. This data can be used at a later time to help increase the reliability of the system and can also be used to infer more information in cases of emergency. For example: using data regarding the current working status of sensors along with the type of connection they use to connect to the server can be used to provide extra information in the event of a fire. Thus, in case of fire, if a sensor stops working (that doesn't go down in normal course of action), it can be assumed that the door and the region around is on fire. As all sensors will have a backup 4G (wireless) connection to the backup servers, it is also possible to work out if the wired connections are going

down. With enough testing this can also be used to work out which parts of the building have been compromised. All of this information can be used to help assist the emergency services.

4.3.2. Physical Machines

There will be at least one server installed on the local network for any implementation. These will be used for mostly data storage and processing. They can also be connected to any alarm systems within the building. Since sensors will check for a connection to the server before sending data, this connection can also be used to define the current operating mode of the system. For example, in normal use the system gets data only once every 60 seconds from each sensor. However, in the event of a fire or any other emergency this slow update time will be problematic and it would instead be better to push the updates live to the server. This will slow down the processing of the data in the database, but it is also set up to change operation slightly within an emergency situation in order to process this data quicker. This is highlighted more in the protocol section 5 in appendix G. All physical machines will back up every 15 minutes in normal use to an off-site system. This system can be used to do larger data analysis, especially in case of emergencies to help predict the movement of people through buildings better; hence improving the accuracy of the system over time. Also, the data can be accessed remotely from the offsite backups in case it cannot be accessed from the local databases. Furthermore, it is planned that sensors will be able to as a backup also send to this off site system if a connection to the local database goes down.

4.3.3. Access

Access to the data will be restricted and needs to be controlled strictly. The system will be installed on its own private and secure Local Area Network (LAN). A tablet or another similar type of device will be stored with either building security or in the reception office that will also have access to this secure network. This device will be able to directly access the server that will display the data on a webpage that will be similar to the concept shown below:



Figure 9: Concept art for the user interface

The interface to the server is made up of a floor plan displaying coloured dots indicating the number of people in an area. It also displays a list of the occupied rooms and their room count. If there are any errors detected they appear in the error log. The interface is broken down into sector views, occupied room lists and floor plans. This allows easy access to a visual distribution of the people in the building, as per the requests of the clients which is what the clients requested, (See appendix F). The error log and occupancy list are designed for use by building managers and security, as they provide useful information to these users. The heat map is generated by placing coloured dots in rooms that are occupied. The colour break down is red for occupancy \geq 8 people, orange \geq 4 people, yellow > 0 people.

4.4. Hardware implementation

The hardware implementation revolves around the door setup. In a real implementation, the system would be mounted to an existing door frame, which has been replaced by an aluminium frame for the purposes of the demonstration.

In order to imitate the parameters of a real door, the aluminium frame was chosen to be 2m in height by 1m in width. Aluminium profiles were used for easy implementation of the same.



Figure 10: Aluminium frame cross section - http://uk.rs-online.com/web/p/tubing-struts/7613284/

The laser mounts were provided by a Dr Adrià Junyent-Ferré, 3D printed, and then attached to the aluminium frame at a height of around 1 metre (110 cm and 125 cm) from the ground. These mounts hold the 4 lasers placed at the vertices of a square that encloses the frame between its two vertical sides. The mounts allow adjustment of the laser in two planes, making it convenient to direct the laser to the phototransistor from one end of the door to the other, while holding the lasers firmly in place.

The order in which these lasers detect the person will define the direction in which the person crosses the door. Also, having lasers at two different heights enables accurate calculation of the number of people crossing through the door, in case of variations in heights in different people. The use of two pairs of lasers also makes the system robust, as explained previously, in case of one of the lasers being compromised.

The webcam that is used for the Computer Vision (CV) is attached to the top bar of the frame for the purposes of the demonstration, with a wide angle lens attached to it. Although, in real life implementation, it would be ideal to attach the webcam to the ceiling, in order to get a wider view.

4.5. Connecting the systems for implementation

For the CV and laser systems to work concurrently, Python's threading module (8) had to be researched. This module allows the execution of the CV and laser system code in parallel in separate threads. The main thread runs the code to send the data through SQL queries. The main thread also contains the instance of the People class, count, which contains the amount of people in the room from both the laser and CV systems as well as getter, setter, and reset functions. The class also contains previous values of the room count, and a function to copy the current values to previous. Locks are used to synchronise the laser and CV threads. Both threads will initialise the resources they need, then wait until the main program thread is ready and sets the flag allowing them to run. This is done via Threading's multiple Event object, which is а simple object used to synchronise threads.



Python's implementation of threading allows all the threads to access the shared resources from the main program thread. This is utilised such that the CV and laser threads can write to the same instance of the *People* class, making it simpler to send to the database. When the CV or laser thread detects a person, it calls the corresponding setter function in *count* and causes the appropriate change to the values stored in *cv/laser* (variables in the class). This change is accessible to the *sqlConnectSend* function that runs in the main thread. The separate laser and CV counts are included so that the discrepancies between the two counts can be tracked to test the relative accuracy of both system. In the actual implementation, both counts still exist to allow the CV and laser to work independently allowing for redundancy in the case of a failure of one system.

However in the situations such as narrow doors with low ceilings, the CV system would be less suitable. Since no more than one person passing through the door at a time, the laser system would be suitable to accurately detect people's movements through this doorway. Whereas in the case of a wide doorway, where more than one person would be able to pass through, CV would be better at doing this due to being able to track more than one person moving at a time. Therefore detecting this instance, whereas the lasers would only detect one person.

The main program initialises the connection to the server prior to allowing the execution of the laser and CV threads. This verifies the availability of the server and then closes the connection. The laser and CV threads then continue. From testing, connecting to the sql server takes approximately 10 seconds. Every 50 seconds (so data is sent approximately every minute), the *sqlConnectSend* function is called. This connects the pi to the server. If it is connected, it will get the current and previous values of the counts and get the change for both laser and CV (current - previous). Then it sets the previous value to the current value, and sends the change in the number of people in the room (along with the respective IDs) to the CV and laser database. If it is unable to connect, nothing is sent and the previous value is not updated so that no data is lost. It then tries again in approximately 60 seconds. As this process takes place, the CV and laser threads still keep running and counting people. The database then updates the value of the people in the room accordingly.

The final code used for the whole system is included in appendix E.

Testing of the complete system was conducted. Both systems were accurate to over 85%, with CV hitting 90% in the complete testing. CV accuracy was limited by the low height of the webcam used, as this caused people to occupy most of the camera's field of vision. This issue means that when used at its current height, the webcam, is unable to count more than one person passing through the doorway. However, the doorway for the demonstration is not big enough to support two people walking through it simultaneously so this was not an issue for the purposes of demonstration. With a higher camera mount, CV's accuracy would increase, as people would be smaller objects and background reduction would work better resulting in smaller contours to track. The laser system worked and updated faster than the CV system because it is computationally less intensive. Testing revealed each loop of the CV code was taking ≈ 250 ms to run, whereas the laser code was practically running every 1 ms plus the slight overhead of the FSM code. The CV code was limited by the processing power of the Pi. This could

be improved by optimising the algorithm used, as well as by increasing the priority of the thread running it to give it more CPU time.

In the final implementation, the code would be modified to deal with errors more robustly (exception catching), and would be modularised. The modularisation would allow one codebase for all the possible variations of the system (CV, multiple doors connected to one controller etc.) and simply using python's import ability. Alternatively the system would be redone in C++, as all libraries used exist as C++ variants.

5. Meeting design criteria

5.1. Social Impact

This project completes its goal of engineering for social impact as it can be used to save lives during an emergency. As the system allows for rescuers to be made aware of where people are in the building they will more easily be able to find them in time to save them. In addition, rescuers may not have to enter a dangerous location if they know that there is nobody there. The system also enables security teams to monitor movements through buildings and hence can serve as an anti-crime device, by spotting and identifying intruders in case of a break in. Other social applications include monitoring traffic flow through buildings so that architects can design better, more effective buildings.

5.1.1. Emergency Responders

In the event of an emergency, for example a fire, responders try to establish if there are any people left in the building and where they are. LifeCounter will help them conveniently find this information and create a plan to reach and rescue them. This innovative system will help save lives and improve the quality of life for people who are trapped in fires but can be located more quickly than before. Additionally, LifeCounter will advise emergency responders whether or not they need to enter certain areas, therefore preventing any injuries to the responders themselves. The risk is further reduced to responders as they will be able to locate victims quicker, and hence will be exposed to danger for a shorter duration of time.

5.1.2. Security Teams

One major issue with large buildings and sites with many different buildings is that it is hard to tell if there is someone trespassing on the property without having many people on patrol or looking at camera feeds. LifeCounter will be able to provide quick, live information regarding the location and number of people on-site. This will effectively help identify if there is an unauthorised person on-site.



5.1.3. Building Managers & Architects

LifeCounter will provide very useful information to building managers and architects as it will be able to keep track of the motion of people throughout the building and from this establish trends in how people are moving. This information can be used to design buildings that better suits the way people use the space or make modifications to routes through existing buildings.

5.2. Design Criteria Check

Specific Point	Pass / Fail
PWR 1.1	Pass. The system is powered by a 5V adapter connected to a 230V mains supply
PWR 1.2	Fail. As it stands the system cannot be powered by PoE, however it is relatively simple to implement.
PWR 2	Fail. No SPS exists as of yet.
PWR 3	Pass. The system does not draw more than 60W.
DEV 4.1	Pass. The system is able to determine direction.
DEV 4.2	Fail. The system is accurate to approximately 90%
DEV 4.3	Pass. The cameras are able to detect people of any height below the camera.
DEV 4.4	Pass. The cameras can detect people of a width greater than 35cm.
DEV 4.5	Pass. The cameras are able to cope with multiple people moving simultaneously.
DEV 4.6	Pass. The minimum travel time through the door is 10ms for the system to register the motion correctly.
DEV 5.1	Pass. None of the elements of the system are capable of causing injury so long as they are used appropriately.
DEV 5.2	Pass. The system is easy to install to new and existing buildings without any damage occurring.
DEV 5.3	Pass. No active interaction is required.
DEV 6	Maybe. It depends where the object is left.
DEV 7.1	Fail. The data is pushed to the server at least every 60 seconds, though this was deemed acceptable. In a fire event the data is sent immediately.
DEV 7.2	Pass. The data can be sent over a wireless network.



DEV 7.3	Pass. No extra hardware is required, however a separate subnet mask is required.
DEV 8.1	Unknown.
DEV 8.2	Pass. No new infrastructure is required to mount the system.
DEV 8.3	Pass. The device is attached in a robust way and will withstand mechanical interactions.
LCL 9	Fail. The server will currently accept all data sent to it.
LCL 10	Pass. The local server logs data history.
LCL 11	Fail. The local server does not have any special power features as of yet.
GLB 12	Fail. The global server will accept any and all data sent to it.
IFC 13	Pass. The interface shows a readout of the data in an easy to use way.
IFC 14	Pass. The interface is a one way data transfer portal.
IFC 15	Pass. The interface clearly shows occupancy
IFC 16	Fail. The interface is not yet password protected.
IFC 17	Fail. The interface does not yet display all of this information
IFC 18	Pass. The interface works on many different devices.

Table 4: Design Criteria checklist

5.3. Cost of System

Appendix I shows the cost break down of the prototype system. As some components were already owned by members of the team, they are not included in the purchase report.

6. Project management

Managing the project through its development was done via regular meetings, splitting into teams and ongoing group discussions via Slack. Using the teams to develop the system individually means that all areas of the system could be designed in parallel. Furthermore regular project meetings and the appointment of a project leader to oversee the progress meant that all internal deadlines were met. Please see Appendix A for the meeting minutes and Appendix B for the project plan. The project plan has been regularly updated as the system has been developed.



7. Future Work

Further development of the system would allow for the design to be improved. This section details the changes to the design that would be made if more time were available.

- Use of infrared emitters and receivers instead of visible red lasers, as they will be less powerful hence safer, and the system will be less prone to human interference
- An array of emitters and receivers will provide more accurate results compared to the ones provided by pairs of emitters and receivers that are currently in use
- The algorithm used in CV detection can be improved in order to take occlusion into account
- Routinely checking the sensors to make sure they are unobstructed
- Adding the ability for the Pi to turn the lasers on and off to verify the sensors are working and allow for auto calibration of the system
- Instead of being a tabulated representation of data, the data can be graphically represented on the blueprints of the building plans, making it easier to comprehend

The scope for future development of the scope of this project is also very promising. Serving as a source of important data to businesses and companies, it can be used for purposes other than just emergencies.

• The server and interface could be developed to show long term trends in traffic flow to building managers, architects etc.

8. Conclusion

LifeCounter is a system that is able to determine how many people have passed through different points in the building, how many people are present in a specific location in the building and other important information regarding the people in the building. From this report it can be seen that it is possible to expand LifeCounter to cover a wider network of locations and provide emergency responder, building managers and security teams with live data regarding the distribution of people throughout the site. The two detection methods work well and coherently to collect data and process it in order to relay it to the server for display on the interface. This interface provides easy access to the data for the end user and allows for occupancy across the site to be monitored. Further steps to develop the system include developing the interface, making the switch from lasers to IR beams, improving the data processing of the computer vision system and moving from off the shelf hardware to custom circuitry to reduce the cost. LifeCounter is the next step technology takes to preventing deaths and injuries caused due to fires and other similar emergencies due to lack of data about the people in different parts of the building / house / complex facing the emergency. By reducing the response time taken by the emergency responders, LifeCounter is a product every building should have in the best interests of its occupants.



9. References

- (1) abyz.co.uk, p. @. (2017, Feb 19). *Python Interface*. Retrieved from <u>http://abyz.co.uk/rpi/pigpio/python.html</u>
- (2) Broadcom. (2012, Aug 2). *GPIO pads control*. Retrieved from Scribd: https://www.scribd.com/doc/101830961/GPIO-Pads-Control2
- (3) Spurrier, N. (2002, Aug 22). *FSM in Python*. Retrieved from noah.org: <u>http://www.noah.org/python/FSM/</u>
- (4) OpenCv. (2016, Dec 23). *How to use background subtraction methods*. Retrieved from Open CV: http://docs.opencv.org/3.2.0/d1/dc5/tutorial_background_subtraction.html
- (5) OpenCV. (2015, Dec 18). *Smoothing Images*. Retrieved from Open CV: http://docs.opencv.org/3.1.0/d4/d13/tutorial_py_filtering.html
- (6) OpenCV. (n.d.). *Eroding and Dilating*. Retrieved from Open CV: http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html
- (7) Alper Yilmaz, O. J. (2004, June). *Object Tracking*. Retrieved from http://vision.eecs.ucf.edu/papers/Object%20Tracking.pdf
- (8) Hastings, L. (2017, Jan 27). *Threading*. Retrieved from Python Standard Library: https://docs.python.org/3.4/library/threading.html



10. Appendix

A) Meeting minutes

Date	Minutes	
22/02/2017	In attendance	
	 Jacob Karmanya Samuel Zheng 	
	Minutes	
	 Discussed report A lot of things need doing Final stage of project - less than 4 weeks to go Report needs starting 	
	Action Points	
	 Email Adria about where he got the frame from CV team need pi from Alex/George Need to design the frame ASAP Needs building ASAP 3D printing the laser mounts from the files Adria sent Can they be adapted to also fit the photo transistor Alternative - Stripboard Report structure needs starting Presentation not urgent, but needs thought Website Alex find out if you can get wordpress installed ASAP So Laura can start design FSM needs implementing Update project plan 	
27/02/2017	Bring CV and laser together. Jacob will sort out SQL. E(x) sent.	
	things.	
	On track to get a working demo.	



	both baby photos and actual ones on the website.
	Report
	Quite simple lots in interim report. worth 45%.
	Write protocols for everything. talk about design having impact on social how innovative??
	Presentation
	-state problem 50% on technical content 30% actual what we are doing 20% make sure everyone knows what the fuck is going on
	ΡΟΑ
	Laura finish up writing up the marks George write up the finite state machine
	Wednesday the 8th of march put everything together. meeting friday at 1
03/03/2017	 Discussion on Dual authentication Will use both, however in reality would use lasers on small doors and CV on large doors Door arriving monday Need to 3D print more mounting parts for the laser - print spare Work on front end website - people to send photos and laura to write content Alex to install python distribution for the server Report writing this weekend - priority over maths report Combining systems on wednesday

B) Project plan

Task	Start Date	End Date	urat on Delegated to	Level of Risk	Risks
Write Project Plan	12/12/2016	16/12/2016	4 Laura	Medium	Too little detail, not able to follow plan; too much detail, too much time spent updating and managing project plan
Research on Raspberry Pi/Audrino/ARM	16/12/2016	09/01/2017	24 George/Omar/Jacob	High	If sufficient research is not completed, the wrong interface will be chosen, affecting the entire project
Research on wireless interface	16/12/2016	09/01/2017	24 Samuel	Medium	If sufficient research is not completed, the wrong interface will be chosen, affecting the entire project
Research on Servers	16/12/2016	09/01/2017	24 Alex	Medium	If sufficient research is not completed, the wrong server will be chosen, affecting the entire project
Research optical sensors	16/12/2016	09/01/2017	24 Laura	Medium	If sufficient research is not completed, the wrong sensor will be chosen, affecting the entire project
Research on dual authentication	16/12/2016	09/01/2017	24 Zheng Yang	Medium	If sufficient research is not completed, the wrong sensor will be chosen, affecting the entire project
Choose the right options for use	09/01/2017	13/01/2017	4 Everyone	High	Not enough evidence is submitted, leading to illinformed decisions. Omittted by thorough research
Start developing dual authentication system	09/01/2017	20/01/2017	11 Omar/Samuel/Zheng	Medium	Too many resources going into the second system, so that the first system doesn't work as well
Research on Lasers	11/01/2017	13/01/2017	2 Kamanya	Medium	Safety research on lasers was required along with a cost approximation and to see compatibility with Arcluin
Developing the laser code	14/01/2017	21/02/2017	38 Jacob/George	Medium	Could be very time consuming
Developing the CV code	14/01/2017	21/02/2017	38 Samuel/Zheng Yang	Medium	Could be very time consuming
Start cost analysis of the project	14/01/2017	18/01/2017	4 Alex	Medium	Unable to give an accurate estimate at this point in the project
Start sensor testing(CV)	14/01/2017	20/01/2017	6 Samuel/Zheng Yang	Medium	If tests aren't thorough, the wrong sensor layout wont be chosen, impacting future functionality of the project
Database designing	18/01/2017	10/02/2017	23 Alex	Medium	Creating the wrong type of database would mean the rest of the project is directly affected
Design interim report structure	18/01/2017	22/01/2017	4 Laura	Medium	Mark scheme is not out for report, therefore what is needed is not known
Start drafting interim report	18/01/2017	25/01/2017	7 Everyone	Low	If report is not drafted then group input is limited and time pressure is more apparent
Start sensor testing(Laser)	20/01/2017	28/01/2017	8 George/Jacob/Karmany	a Medium	If tests aren't thorough, the wrong sensor layout wont be chosen, impacting future functionality of the project
Website design	20/01/2017	24/01/2017	4 Laura	Medium	People getting too involved in the website that it engulfs the entire project
Make interim deadline plans for individuals	20/01/2017	23/01/2017	3 Laura	Medium	Checking people are on track is tasking, if the checking system is not usable then the project may lose its timings
Conduct market research	23/01/2017	27/01/2017	4 laura	Madium	Market recearch with firefinithers could rive us data to show we are anornachian this incorrectly
	2001000	2100/00/00	A Variation		mannes reaction more mentioners acousting the second second mean and the subproton grant mean coup I also amined of the largest could defaultion the traction of the surface and the amined have lead times accelering to
	Z8/01/2017	1102/20/20	4 Matmanya		באור אווועא מו עוד ואפרוג סטטוס טרואל עדי ובאנווע מו עוד אלאנאיד אווים גיויד ארט איני איני אוויד כטווגעאוווג
Group review of interim report	31/01/2017	31/01/2017	1 Everyone	Medium	Conflicting views and writing styles affect the reports fluidity
Final interim report writing	01/02/2017	08/02/2017	7 Everyone	Medium	Report is not written quickly enough to emit deadline stress levels
Creating private network and servers for the databases to run on	07/02/2017	13/02/2017	6 Alex	Medium	work on a test development wont necesarly work in a real deployment
Submit Interim report	08/02/2017	08/02/2017	1 Laura	Low	Report is not submitted on time, omitted by reminders and calendar entries
Run tests and simulations on the code with sensors	12/02/2017	13/02/2017	1 Jacob	Medium	Tests are not thorough enough to give conclusions
Bring together the sensor data and code	12/02/2017	15/02/2017	3 Jacob	Low	Code cannot adapt to the sensor data
Run tests on accuracy of the sensor triggering	15/02/2017	21/02/2017	6 George/Jacob	Medium	Tests are not thorough enough to give conclusions
Select template for the website and start drafting corperate text	15/02/2017	18/02/2017	3 Alex/Laura	Medium	Website design template is not able to fulfil the needs of the website and it needs to be changed later
Further develop the costing analysis of the project	28/02/2017	02/03/2017	2 Everyone	Medium	Conflicting views on expenses for the high end components delay the analysis
Developing the Pi-Server interface code	01/03/2017	08/03/2017	7 Jacob/George/Alex	Medium	Could be very time consuming
Build door frame for testing and demo	06/03/2017	10/03/2017	4 Omar	Medium	Without a successful testing system there will be no demo
Print laser mounts for the door	08/03/2017	10/03/2017	2 Kamanya	Medium	Without a successful testing system there will be no demo
Start drafting final report	01/03/2017	08/03/2017	7 Everyone	Low	If report is not drafted then group input is limited and time pressure is more apparent
Set up all items for demo	08/03/2017	13/03/2017	5 Everyone	Medium	Project is not working at this point, delaying preparation for final demo
Group review of final report	08/03/2017	10/03/2017	2 Everyone	Medium	Conflicting views and writing styles affect the reports fluidity
Rework draft report	10/03/2017	13/03/2017	3 Everyone	Medium	Report is not written quickly enough to emit deadline stress levels
Combine CV and Laser codes	10/03/2017	13/03/2017	3 Jacob	Medium	If the codes do not integrate properly then work done on both systems is redundant
Run tests on dual authentication system	13/03/2017	15/03/2017	2 Omar/Samuel/Zheng	Medium	Tests are not thorough enough to give conclusions
Analyse and evaluate the effectiveness of the project	13/03/2017	13/03/2017	1 Everyone	High	Conflicting views on what classes as effectiveness mean analysis not all relevant.
Website launch	13/03/2017	13/03/2017	1 Alex	Medium	Website launch is not effective or on time, affecting the project mark unproportionally
Submit Final Report	13/03/2017	13/03/2017	1 Laura	Low	Report is not submitted on time, omitted by reminders and calendar entries
Preparation for presentation and demo	13/03/2017	21/03/2017	8 Everyone	Medium	There are conflicting views in the team, meaning the preparation is delayed
Presentation and demo	21/03/2017	21/03/2017	1 Everyone	Medium	Presentation and demo do not go to plan and project mark is unproportionally affected









C) Computer Vision Code

```
import numpy as np
import cv2
count = 0
im = cv2.VideoCapture(0)
fgbg = cv2.createBackgroundSubtractorMOG2(varThreshold=95) # background reduction function
found = 0
while(found == 0): # loop until first set of contours are found
       ret, old frame = im.read() # capture a frame from the camera
       old_fgmask = fgbg.apply(old_frame) # apply background reduction
       old_retval, old_threshold = cv2.threshold(old_fgmask, 10, 255, cv2.THRESH_BINARY)
       # convert the mask generated by the background subtractor into binary image
       old_median = cv2.medianBlur(old_threshold,55) # reduce noise
       old_median = cv2.dilate(old_median, None, iterations = 3) # connect gaps around the blobs
       old_median = cv2.erode(old_median, None, iterations = 3) # shrink the blobsq
       # find first contours of moving objects
       image, old contours, hierarchy = cv2.findContours(old median,cv2.RETR TREE,cv2.CHAIN APPROX SIMPLE)
       if len(old contours) > 0:
       try: hierarchy = hierarchy[0]
       except: hierarchy = []
       # draw a rectangular box bounding each detected contour
       for contour, hier in zip(old_contours, hierarchy):
              (x,y,w,h) = cv2.boundingRect(contour)
              if (w > 140) and (h > 140): # only a contour with w and h both greater than 140 will be
considered valid
              # find the centroid of the rectangular box
              old x1 = w/2
              old_y1 = h/2
              old cx = x+old x1
              old_cy = y+old_y1
              old_centroid = (old_cx,old_cy)
              found = 1 # update found and terminate the loop
while(1):
       ret, frame = im.read()
       fgmask = fgbg.apply(frame) # apply background reduction
       retval, threshold = cv2.threshold(fgmask, 10, 255, cv2.THRESH BINARY)
       # convert the mask generated by background subtractor into binary image
       median = cv2.medianBlur(threshold,55) # smoothen the blobs
       median = cv2.dilate(median, None, iterations = 3) # connect gaps around the blobs
       median = cv2.erode(median, None, iterations = 3) # shrink the blobs
       # find contours of moving object
       image, contours, hierarchy = cv2.findContours(median,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
       # draw a line on the frame
       line = 240
       cv2.line(frame, (0, line), (660, line), (0,255,0), 4) #draw line
       if len(contours) > 0:
       try: hierarchy = hierarchy[0]
       except: hierarchy = []
       # draw a rectangular box bounding each detected contour
```

```
for contour, hier in zip(contours, hierarchy):
               (x,y,w,h) = cv2.boundingRect(contour)
              if (w > 140) and (h > 140): # only a contour with w and h both greater than 140 will be
considered valid
              cv2.rectangle(frame, (x,y), (x+w,y+h), (255, 0, 0), 3)
              # compute the centroid of the rectangular box
              x1 = w/2
              y1 = h/2
              cx = x+x1
              cy = y+y1
              centroid = (cx, cy)
              cv2.circle(frame,(int(old_cx),int(old_cy)),2,(255,0,0),-1) #show coordinates of previous
centroid
              cv2.circle(frame,(int(cx),int(cy)),2,(0,0,255),-1) #show coordinates of current centroid
              # this condition only occurs if the distance between
              # the current centroid and the old centroid is less than 300
              if abs(cy - old_cy) < 300:</pre>
                      if cy > line: # direction: DOWN
                      if old_cy < line:</pre>
                             if h^*w > 300000: # if area of rectangular box is greater than 300000, count += 2
                             count += 2
                             else:
                             count += 1
                      if cy < line: # direction: LEFT</pre>
                      if old_cy > line:
                             if h^*w > 300000: # if area of rectangular box is greater than 300000, count -= 2
                             count -= 2
                             else:
                             count -= 1
              #update centroid
              old_cy = cy
              old_cx = cx
       # write number of count
       font = cv2.FONT_HERSHEY_SIMPLEX
       cv2.putText(frame, str(count) ,(320,430), font, 1, (255,0,0), 2, cv2.LINE_AA)
       cv2.imshow('frame', frame) # show image captured by camera
       cv2.imshow('median', median) # show background reduced image
       k = cv2.waitKey(1) & 0xFF
       if k == ord('q'):
       break # break the loop when q is keyed in
im.release()
cv2.destroyAllWindows()
```



D) Laser Safety



LASER SAFETY

- ALL PERSONS INTENDING TO WORK WITH LASERS <u>MUST</u> CONSULT THE DEPARTMENTAL LASER SUPERVISOR (DLS) BEFORE STARTING WORK
- ALL PERSONS INTENDING TO WORK WITH LASERS <u>MUST</u> COMPLETE THE COLLEGE "INTRODUCTION TO LASER SAFETY" COURSE

Departmental Laser Supervisor: Dr Andrew Holmes Room 701 Ext, 46239 a.holmes@imperial.ac.uk

When working with Class 2 or 3R lasers [typically < 5 mW HeNe (visible) alignment lasers]:

- Do not stare into beam
- Avoid setting up beam at eye level
- · Terminate all beams with opaque stops within limits of experimental area
- · Direct beam away from populated areas and entrance doors where possible
- Avoid accidental reflections from watches, belt buckles, jewelry and other equipment
- Ensure laser is mechanically secure, and electrically tested to College standards
- Ensure laser displays correct labels according to latest BS EN 60825 standard, particularly black-on-yellow triangular "laser starburst" sign.

When working with Class 3B or 4 lasers

[3B (typically from 5 mW to 500 mW and most infrared lasers): hazard from direct beam viewing and specular reflections]

[4 (above 500 mW): as 3B, but diffuse reflections may also be hazardous to eyes and skin]

In addition to precautions for class 2 & 3R above:

- Entrance door(s) must display a "DESIGNATED LASER AREA" label, and an illuminated flashing "LASER ON" warning sign. The sign must incorporate a "telltale" repeater neon indicator or equivalent as backup in the event of bulb failure
- · Laser operation must be impossible unless warning light is activated
- · Warning light operation must be checked at least daily during laser operation
- Lasers must conform to the latest BS EN 60825 standard with regard to warning labels, key switches, emission
 indicators and beam shutters, etc (See College Laser Safety Code of Practice)
- Entrance areas immediately inside the door must be screened from a direct view of the lasers
- · Local warning signs must be displayed for any special hazards associated with your laser set-up
- A separate screened area must be provided for personnel seated at computers, desks, etc.
- Any unregistered persons who are required to remain in the lab during laser operation must stay under the strict supervision of the laser operator
- Suitable laser goggles (and hand/skin protection, where appropriate) must be available, and used, when other
 engineering controls such, as screening, etc, are considered insufficient as protection
- Attention must be paid to associated hazards such as electrical, chemical, mechanical, etc (fume ventilation, laser dye handling, capacitor banks, leaking water-cooling tubes, unsecured cables across floors, etc)
- ANYONE WITH A SUSPICION OF LASER BEAM EXPOSURE TO AN EYE MUST IMMEDIATELY SEEK MEDICAL ATTENTION, INFORM THEIR LINE MANAGER AND BOTH THE DLS AND DSO

NEWCOMERS MUST BE STRICTLY SUPERVISED UNTIL FULLY AWARE OF THE DANGERS AROUND THEM!

IF IN DOUBT, SEEK ADVICE FROM YOUR SUPERVISOR, LINE MANAGER OR SAFETY STAFF



E) Complete code for project

```
#CV requirements:
import numpy as np
import cv2
#Laser requirements:
import pigpio
#Concurrent execution and database requirements
import threading
import time
import mysql.connector
#Class for counting People
class People:
                     #Current number in room (CV)
       cv = 0
       laser = 0 #Current number in room (laser)
       prevCv = 0 #Last sent number in room to DB (CV)
       prevLaser = 0; #Last sent number in room to DB (laser)
       def reset(self):#Rests things
       self.cv = 0
       self.laser = 0
       self.prevCv = 0
       self.prevLaser = 0
       def copyToPrev(self):
       self.prevCv = self.cv
       self.prevLaser = self.laser
       def cvIn(self, inpep):
       self.cv += inpep
       print("CV people: " + str(self.cv))
       def cvOut(self, outpep):
       self.cv += outpep
       print("CV people: " + str(self.cv))
       def laserIn(self):
       self.laser += 1
       print("Laser people: " + str(self.laser))
       def laserOut(self):
       self.laser -= 1
       print("Laser People: " + str(self.laser))
       def missed(self):
       print("Invalid Input to laser!!")
#Defining pins for laser
class Pin: #Defining input pins
11 = 17 #left1
12 = 27 # left 2
r1 = 22 #right1
r2 = 23 #right2
#Class for FSM
class ExceptionFSM(Exception):
       """This is the FSM Exception class."""
       def __init__(self, value):
       self.value = value
       def __str__(self):
       return self.value
class FSM:
            __init__(self, initial_state):
       def
       """ This initialises the FSM, inital state is set here.
       Transitions are defined in the state_transitions dictionary.
       The FSM has been pre-implemented in this""
       # Map (input_symbol, current_state) --> (action, next_state).
                                          '0'):(None,
       self.state transitions = {('00',
                                                          '0'), # default transition
```

```
#People going into room
       ('01', '0'):(None,
                            '1'), #increment state
       ('00', '1'):(None,
                            '0'), #decrement state
       ('01', '1'):(None,
                            '1'), #loop within state
       ('11', '1'):(None,
                            '2'), #increment state
       ('01', '2'):(None,
                            '1'), #decrement state
       ('11', '2'):(None,
                            '2'), #loop within state
       ('10', '2'):(None,
                            '3'), #increment state
       ('11', '3'):(None,
                            '2'), #decrement state
       ('10', '3'):(None,
                            '3'), #loop within state
       ('00', '3'):("count.laserIn(count)", '0'), #Final state transition to get person
counted as in
       #People leaving the room
       ('10', '0'):(None,
                           '-1'), #increment state
       ('00', '-1'):(None,
                             '0'), #decrement state
       ('10', '-1'):(None,
                             '-1'), #loop within state
       ('11', '-1'):(None,
                             '-2'), #increment state
       ('10', '-2'):(None,
                             '-1'), #decrement state
       ('11', '-2'):(None,
                             '-2'), #loop within state
       ('01', '-2'):(None,
                             '-3'), #increment state
       ('01', '-3'):(None,
                             '-3'), #decrement state
       ('11', '-3'):(None,
                            '-2'), #loop within state
       ('00', '-3'):("count.laserOut(count)", '0'), #Final state transition to get person
counted as out
       #Error cases
       ('11', '0'):("count.missed(count)", '4'),
       ('10', '4'):(None, '3'),
       ('01', '4'):(None,
                            '-3'),
       ('11', '4'):("count.missed(count)", '0'),
       }
       # Map (current_state) --> (action, next_state).
       self.state_transitions_any = {}
       self.default_transition = None
       self.input_symbol = None
       self.initial_state = initial_state
       self.current state = self.initial state
       self.next_state = None
       self.action = None
       def reset (self):
       """Resets FSM, current state becomes initial state, and input becomes None
       Initial state was set by constructor"""
       self.current_state = self.initial_state
       self.input_symbol = None
       def get_transition (self, input_symbol, state):
       """This returns (action, next state) given an input_symbol and state.
       Firstly checks state_transitions dictionary - specific input
       Then the state_transitions_any dictionary - any input
       Then default_transition - incase undefined
       If still nothing, exception is raised."""
       if (input_symbol, state) in self.state_transitions:
              return self.state_transitions[(input_symbol, state)]
       elif (state) in self.state_transitions_any:
              return self.state_transitions_any[state]
       elif self.default_transition is not None:
              return self.default_transition
       else:
              raise ExceptionFSM ('Transition is undefined: (%s, %s).' %
              (str(input symbol), str(state)) )
                      #Change exception handler???
       def process (self, input_symbol):
```



```
"""Processes inputs given to FSM.
       Uses get_transition to get the transition from current_state and input
       to next state and any action.
       Call this to get stuff to happen
       If the action is not None, the function is evaluated and called.
       .....
       self.input_symbol = input_symbol
       (self.action, self.next_state) = self.get_transition (self.input_symbol,
self.current_state)
       if self.action is not None:
              eval(self.action) #Calls function stored,
       self.current_state = self.next_state
       self.next_state = None
#Laser stuff
def laser():
       pi = pigpio.pi() #Initialises pigpio, starting the script
       pi.set_mode(Pin.l1, pigpio.INPUT) #Sets up pin 17 (left1) as an input pin
       pi.set_pull_up_down(Pin.l1, pigpio.PUD_OFF) #no pull up or down resistors as we are
pulling up/down via external resistor network
       pi.set_mode(Pin.12, pigpio.INPUT)
       pi.set pull up down(Pin.12, pigpio.PUD OFF)
       pi.set_mode(Pin.r1, pigpio.INPUT)
       pi.set pull up down(Pin.r1, pigpio.PUD OFF)
       pi.set_mode(Pin.r2, pigpio.INPUT)
       pi.set_pull_up_down(Pin.r2, pigpio.PUD_OFF)
       syncEvent.wait()
       fsm.reset() #Resetting FSM
       while(1): #Shitty way of doing sampling
       time.sleep(0.001)
       inpt = str( int(not( pi.read(Pin.l1) and pi.read(Pin.l2)))) + str( int(not(
pi.read(Pin.r1) and pi.read(Pin.r2))))
       fsm.process(inpt)
#CV stuff
def cv():
       im = cv2.VideoCapture(0)
       fgbg = cv2.createBackgroundSubtractorMOG2(varThreshold=95) # background reduction
function
       syncEvent.wait()
       found = 0
       while(found == 0): # loop until first set of contours are found
       ret, old frame = im.read() # capture a frame from the camera
       old fgmask = fgbg.apply(old frame) # apply background reduction
       old retval, old threshold = cv2.threshold(old fgmask, 10, 255, cv2.THRESH BINARY)
       # convert the mask generated by the background subtractor into binary image
       old median = cv2.medianBlur(old threshold,55) # reduce noise
       old median = cv2.dilate(old median, None, iterations = 3) # connect gaps around the
blobs
       old_median = cv2.erode(old_median, None, iterations = 3) # shrink the blobsq
       # find first contours of moving objects
       image, old_contours, hierarchy =
cv2.findContours(old_median,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
       if len(old_contours) > 0:
              try: hierarchy = hierarchy[0]
              except: hierarchy = []
              # draw a rectangular box bounding each detected contour
              for contour, hier in zip(old contours, hierarchy):
              (x,y,w,h) = cv2.boundingRect(contour)
              if (w > 140) and (h > 140): # only a contour with w and h both greater than
140 will be considered valid
                      # find the centroid of the rectangular box
```

```
old_x1 = w/2
                      old_y1 = h/2
                      old_cx = x+old_x1
                      old_cy = y+old_y1
                      old_centroid = (old_cx,old_cy)
                      found = 1 # update found and terminate the loop
       while(1):
       ret, frame = im.read()
       fgmask = fgbg.apply(frame) # apply background reduction
       retval, threshold = cv2.threshold(fgmask, 10, 255, cv2.THRESH_BINARY)
       # convert the mask generated by background subtractor into binary image
       median = cv2.medianBlur(threshold,55) # smoothen the blobs
       median = cv2.dilate(median, None, iterations = 3) # connect gaps around the blobs
       median = cv2.erode(median, None, iterations = 3) # shrink the blobs
       # find contours of moving object
       image, contours, hierarchy =
cv2.findContours(median,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
       # draw a line on the frame
       line = 240
       cv2.line(frame, (0, line), (660, line), (0,255,0), 4) #draw line
       if len(contours) > 0:
              try: hierarchy = hierarchy[0]
              except: hierarchy = []
              # draw a rectangular box bounding each detected contour
              for contour, hier in zip(contours, hierarchy):
               (x,y,w,h) = cv2.boundingRect(contour)
              if (w > 140) and (h > 140): # only a contour with w and h both greater than
140 will be considered valid
                      cv2.rectangle(frame, (x,y), (x+w,y+h), (255, 0, 0), 3)
                      # compute the centroid of the rectangular box
                      x1 = w/2
                      y1 = h/2
                      cx = x+x1
                      cy = y+y1
                      centroid = (cx, cy)
                      cv2.circle(frame,(int(old_cx),int(old_cy)),2,(255,0,0),-1) #show
coordinates of previous centroid
                      cv2.circle(frame,(int(cx),int(cy)),2,(0,0,255),-1) #show coordinates
of current centroid
                      # this condition only occurs if the distance between
                      # the current centroid and the old centroid is less than 300
                      if abs(cy - old_cy) < 300:</pre>
                      if cy > line: # direction: DOWN
                             if old cy < line:</pre>
                             if h*w > 300000: # if area of rectangular box is greater than
300000, count += 2
                                     count.cvIn(count, 2)
                             else:
                                     count.cvIn(count, 1)
                      if cy < line: # direction: LEFT</pre>
                             if old_cy > line:
                             if h*w > 300000: # if area of rectangular box is greater than
300000, count -= 2
                                     count.cvOut(count, 2)
                             else:
                                     count.cvOut(count, 1)
                      #update centroid
                      old cv = cv
                      old cx = cx
       # write number of count
```



```
font = cv2.FONT_HERSHEY_SIMPLEX
       cv2.putText(frame, str(count.cv) ,(320,430), font, 1, (255,0,0), 2, cv2.LINE_AA)
       cv2.imshow('frame', frame) # show image captured by camera
       cv2.imshow('median', median) # show background reduced image
       im.release()
       cv2.destroyAllWindows()
#sql stuff
def sqlConnectSend():
       try:
       cnx = mysql.connector.connect(user='LifeCount', password='admin123',
                                            host='192.168.0.101', port="3306",
                                            database='LifeCounter')
       except mysql.connector.Error as err:
       if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
              print("Error with username or password")
       elif err.errno == ER_BAD_DB_ERROR:
              print("Database doesn't exist")
       else:
              print(err)
       if not(cnx.is_connected()):
       print("Not connected")
       try:
               cnx = mysql.connector.connect(user='LifeCount', password='admin123',
                                    host='192.168.0.101', port="3306",
                                    database='LifeCounter')
       except mysql.connector.Error as err:
               if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
              print("Error with username or password")
              elif err.errno == ER_BAD_DB_ERROR:
              print("Database doesn't exist")
              else:
              print(err)
       cursor = cnx.cursor()
       send cv = ("INSERT INTO PeopleCountCV"
               "(SensorID, RoomID, CountChange) "
               "VALUES (%s, %s, %s)")
       send_laser = ("INSERT INTO PeopleCountLasers"
               "(SensorID, RoomID, CountChange) "
               "VALUES (%s, %s, %s)")
       if(cnx.is connected()):
       print("Is connected")
       laserChange = count.laser - count.prevLaser
       cvChange = count.cv - count.prevCv
       count.copyToPrev(count)
       laserData = ('000001000001', '000001', str(laserChange))
       cvData = ('000001000001', '000001', str(cvChange))
       cursor.execute(send cv, cvData)
       cursor.execute(send_laser, laserData)
       cnx.commit()
       else:
       print("Error with connection, not transmitting change")
       cursor.close()
       cnx.close()
#Initialising FSM
fsm = FSM(initial_state='0')
#initialising count as instance of People
count = People()
#initialising the two threads
cvThread = threading.Thread(target=cv)
laserThread = threading.Thread(target=laser)
```



#Synchronsiing event: syncEvent = threading.Event() syncEvent.clear() count.reset() #Resetting count #Starting the two threads laserThread.start() cvThread.start() #Testing connection to server sqlConnectSend() #Main program loop: while(1): time.sleep(50) sqlConnectSend()



F) Market research - Email Conversation with the fire service

Hello,

Further to our discussion on the phone on Thursday, I have put together a list of questions that would be really useful for our project if we got your opinion on. It would be beneficial even if the answers are that you don't think it would be used.

The project we are working on is building a system to reliably count people= inside a building and where they are in the building. This would be displayed on an easy to access website.

1) Would the fire service have time to look at a system like this to see where people are if the building was on fire?

2) What would be the best way to access this information? On a tablet/mobile device app or through a website for example

3) How would you want the information displayed? In a map or list format for example?

Thank you very much for your help,

Kindest regards,

Laura Tuckey

26 Jan (12 days ago)

KIM.JERRAY-SILVER@london-fire.gov.uk

Hey Laura,

Many thanks for putting your questions into email form, I will do my best to answer them :o)

Question 1:

Would the fire service have time to look at a system like this to see where people are if the building was on fire?

Answer:

As you can appreciate time is paramount when it comes to saving lives.



When we turn up to a fire in a building many thoughts and questions during our information gathering stage need answering?

Normally when? Where? Why? Who? and how?

The faster +can get these questions answered the faster we can commit fire crews to the effected area.

What works extremely well for us is having an on site manager, duty officer or responsible person meeting us on our arrival with building plans showing us the exact location of the fire, risks that may harm us and the location of any people involved / in need of rescue.

If we have no plans we have a systematic approach starting to search the effected area first and making our way throughout the building from there.

Your idea of letting us know exactly where those people are will of course speed up our search process and avoid searching areas unnecessarily.

So in answer to question 1, yes we would have time and would allocate resources to gather this great information that you have provided.

Question 2: What would be the best way to access this information? On a tablet/mobile device app or through a website for example.

Answer:

As we carry no online technology this would be the hardest part of your great idea.

What the Brigade work with are Premises information boxes where plans, keys, building facility instructions such as mechanical ventilation systems are stored and can only be opened with a Girda key which all frontline fire engines carry. They look the attached document. We do not carry mobile phones or lpads as yet.

Our 1st port of call when entering a building is to the automatic alarm panel or control centre where we can see which area/zone has been actuated by reading the LED display. These are almost always located at the main entrance to the building.

Any information you would like to provide for us is best stored near to this panel either on the wall or within a folder.

If it was something like an Ipad or something of value the safest place for this would be within the Premises information box as I explained above.

We can make a note of any pre planned information on our basic information monitor called a mobile data terminal which is on the fire engine but this is purely to say that further information for this incident/address is stored in your premise information box.

We can also make a date to visit the site and familiarise ourselves with your particular premises information box or other systems that you have in place.

Unfortunately our on board information monitor does not have the technology to store your idea on.

Question 3:

How would you want the information displayed? In a map or list format for example?

Answer:

The best format for us would be in map form as we are very used to drawing plans/maps and reading them.

I hope that this answers your questions Laura, If you need any further information or for me to explain in more detail any of the areas listed above please feel free to get back to me.

Many thanks and good luck with what sounds like a great project,

Kim. :o)

Watch Manager Red Watch Acton Fire Station (G26) 0208 555 1200 Ext 84726

Email disclaimer

The information in this email may contain confidential or privileged materials.

Please read the full email disclaimer notice at <u>london-fire.gov.uk/EmailDisclaimer</u> <<u>http://www.london-fire.gov.uk/EmailDisclaimer.asp</u>>

For fire safety advice please go to <u>london-fire.gov.uk/YourSafety</u> <<u>http://www.london-fire.gov.uk/YourSafety.asp</u>>

G) Database protocol

- 1) The database will consist of 3 main sections
 - a) Current Data
 - i) This will be data yet to be applied to the room counts
 - ii) Stored in tables called PeopleCountxxx
 - (1) Where xxx is the sensor name
 - iii) There will be the same number of tables as there are different sensor types.
 - b) History
 - i) This will store all the data collected.
 - ii) Stored in tables called PeopleCountHistoryxxx
 - (1) Where xxx is the sensor name
 - iii) There will be the same number of tables as there are different sensor types
 - c) Processing
 - i) These tables will store all the information that will be used for outputting information.
 - ii) This includes the Rooms table which stores the total count for each room.



- iii) As well as the tobefound table which stores the sensors and which rooms they are connected too.
- iv) Finally it also stores all the sensor information.
- 2) Data will only ever be inserted into the database, it will never be updated by external sources.

3) Rooms will only ever be updated and will never have data inserted or deleted from an external source.

- 4) Normal Use:
- a) Data is inserted into the respective tables by the sensors.
 - i) This data includes:
 - (1) SensorID
 - (2) RoomID
 - (3) Time
 - (4) CountChange
- b) Data is accessed through a website
 - i) The people count is accessed through the local network via a webpage
 - (1) The webpage pulls data from the Rooms table
 - (2) There will be multiple ways of displaying this
 - (3) Data will be displayed solely via requests
- 5) In an emergency
- a) Data will be pushed to the server rather than sent every 60 seconds
- b) The server will not perform any more than a simple analysis on the data.
 - i) Simple analysis assumes that the data from a given sensor is right and ignores other sensors on a door.
- c) After the flow of data to the server has slowed further analysis will commence
 - i) This means that information will become more accurate with time after an incident
 - ii) Allows for a good initial estimate with even better estimates of where people are as time progresses.
 - iii) When an emergency is triggered, depending on the type different path finding algorithms will be used if it is known where people should be heading to.

H) Design Specification

Definitions (DEF)

- 1. LifeCounter is made up of three sub systems,
 - i. The detection and processing unit, referred to as the *device*,
 - ii. The server which collects data from all the devices in a building, referred to as the *local server*,
 - iii. The server which collects data from all the local servers and collates prepares them for viewing, referred to as the *global server*,
 - iV. The online front end for data display, referred to as the *interface*
- 2. Nominal Operation is defined as
 - i. Detecting the movement of people past the device with the same rate of success as in



ideal conditions,

- ii. Communicating data from the device to the local server with the same rate of success as in ideal conditions,
- iii. Communicating data from the device to the local server within 150% of the time taken to do so as in ideal conditions.
- 3. Where Ideal Conditions are defined as
 - i. Room temperature (approx. 20 degrees Celsius),
 - ii. The device is powered from its primary power supply. See PWR-1.
 - iii. The building in which the device is located is experiencing low people traffic.
 - iV. The building in which the device is located is not on fire.

Design Specification

Power (PWR)

- 1. The device must be able to be powered from standard power means within an office building. This is further referred to as the *Primary Power Supply (PPS)*. This includes but is not limited to
 - i. 230V mains electrical supply,
 - ii. Power over Ethernet (PoE) by using installed CAT5 Ethernet cable runs.
- 2. The device must have some self power system.
 - i. This system will allow for nominal operation in the event of power failure to the device from the PPS. This is further referred to as the *Secondary Power Supply (SPS)*.
 - ii. The SPS must be able to maintain nominal operation of the device for at least 60 minutes.
 - iii. The SPS must be able to recharge from the PPS in the event of depletion of SPS followed by reconnection to the PPS.
 - iv. The SPS must be replaceable without significant difficulty if the SPS fails.
 - V. The device should notify the user if the SPS is close to running out of charge.
- 3. The device must not draw more than 60W from the PPS.

Device (DEV)

- 4. The device must be able to detect the passing of a person through the doorway.
 - i. The device must be able to detect which direction the person is passing in.
 - ii. The device should have an accuracy rate of greater than 95%.
 - iii. The device must be able to detect the motion of a person of a height greater than 56.4cm.
 - iV. The device must be able to detect the motion of a person of a shoulder width greater than 35cm.
 - V. The device should be able to detect multiple people using the same entrance / exit to a room simultaneously.
 - vi. The device should be able to uniquely count the number of people using a doorway for



rates of up to 2 people a second.

- 5. The device must used a detection method that
 - i. Is completely safe for use through all reasonable interactions with people,
 - ii. Does not cause damage to the building in which it is installed,
 - iii. Does not require active interaction from people in order to register their movements.
- 6. The device should be able to maintain normal detection accuracy even if an object is left in the doorway.
- 7. The device must communicate data to the the local server.
 - i. This should happen within 30 seconds of the detection of a person passing through the door.
 - ii. Data communication to the local server should occur wirelessly.
 - iii. The device should use existing wireless infrastructure to transmit the data to the local server.
- 8. The device must be robust.
 - i. The device should be able to withstand temperatures of 176 degrees Celsius for 4 minutes.
 - ii. The device should be able to be easily attached to existing infrastructure (door ways etc.) with minimal time and effort.
 - iii. The device should be able to withstand mechanical interactions with people, for example being kicked or walked in to, and continue to function correctly.

Local Server (LCL)

- 9. The local server must reject data not sent from a device registered with the system.
- 10. In the event of the local server loose connection to the global server, the local server should log all data sends from the devices on its network until connectivity has been restored to the global server.
- 11. Specification points PWR-1, PWR-2 & PWR-3 and their sub points all apply to the local server.

Global Server (GLB)

12. The global server must reject data not sent from a local server registered with the system.

Interface (IFC)

- 13. The interface must facilitate access to the global server in an easy to use way.
- 14. The interface should not allow for data in the global server to be changed through the interface.
- 15. The interface must clearly show the number of people in each room in the building.
- 16. The interface must only display data to authorised users.



- 17. The interface should allow for critical data about the device(s) on the system to be shown to the user, including but not limited to
 - i. The battery level of the SPS,
 - ii. Connection strength of the wireless network,
 - iii. If the device is powered from it's PPS or SPS,
 - iV. Any critical internal errors or faults found on by the device.
- 18. The interface must work the same on a mobile device (smartphone, tablet etc.) as it does from desktop.



I) Cost Breakdown

Raspberry Pi	
USB Extension	
Aluminium Components for door frame	
Mechanical components for door frame	
3D printing for door frame attachments	
TOTAL	£100.78

As can be seen, the majority of the cost comes from the door frame. This is not a cost required by the system, but is for testing.