University of Glasgow | School of Computing Science

Who's There? Occupancy Prediction with Smart Sensor Boxes

Jamie Sweeney

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

Level 4 Project — September 26, 2017

Abstract

We show how to produce a level 4 project report using latex and pdflatex using the style file l4proj.cls

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: ____________________ Signature: ______________________

Contents

Chapter 1

Introduction

1.1 Context

The internet of things, or the IoT is a term used to describe a wide network of connected devices, ranging from supercomputers to smart appliances. Since the term was first coined in 1999 [1], the IoT has come a long way with new applications being discovered consistently [source needed], with a predicted market of 6.5 trillion) by 2020 [1].

One key area of study has been developing occupation detection that would allow IoT devices to be more aware of the occupation status of their surroundings. This is a crucial step to implementing smart homes, workplaces and campuses that are responsive and can meet their demands at any given time.

Advanced examples are already taking form, with the University of Glasgow investing 1bn towards developing a smart campus and the city of Barcelona making major overhauls to many public services in an attempt to bring the idea of a smart city to reality. Goals The aim of this project is to design and construct low budget smart sensor boxes, capable of predicting human occupation in a room. To do this we use the popular brand of microprocessors Raspberry Pis in addition to a variety of sensors and other techniques.

In addition to the smart sensor boxes, we will design and implement a centralised hub capable of receiving, storing and serving information produced by a series of RPi boxes. This hub should also provide functionality for system-wide maintenance and configuration, to make setup and operation efficient and flexible.

1.2 Motivations

The motivations for this project are wide, with smart sensors being applicable in a broad variety of ways. We will discuss the primary motivations for this project, which are understandably more relatable and accessible to myself, in addition to a number of secondary motivations that are considered due to close similarity.

1.2.1 Primary Motivation

To narrow our field of focus, we choose a main motivation to consider when producing the smart sensor boxes. We consider the following scenario: University administration decide they wish to upgrade buildings with smart sensor boxes with the overall goal being that student or other staff member can find current/past occupancy information on certain areas remotely. For example library group study areas could be listed online with an indication of their current use, allowing students who plan on visiting to check availability beforehand. The University of Glasgow library currently provides a similar feature, displaying computer use breakdown by floor, available from monitors within the library.

In addition the system should be built to be easily extendable, and capable of operating as a part of a larger IoT network.

1.2.2 Secondary Motivations

Although we focus on a university implementation of occupancy prediction, we can imagine a similar design could be incorporated into many different areas. Schools, workplaces and public buildings could benefit from installing smart sensor functionality, allowing not only occupancy information to the user, but also providing more in-depth analysis of overall occupation to any interested administrators.

An example of this would be public buildings that allow local councils to analyse the usage of public service, allowing them to make more informed decisions on funding, maintenance, staffing and other key elements of successful operation.

Smart sensor data would come in useful for many other functions, perhaps one of the most useful being used as supplementary information to existing systems. For example automated heating, ventilation, lighting and other system functions could benefit from the additional knowledge and perhaps function more independently as a result.

To list a few more: Security systems, home-automation, public transport, gyms, dining locations and other areas of leisure.

Chapter 2

Background Analysis

We start by researching and reviewing past studies conducted in similar fields, given that the subject matter is broad and widely applicable, we find many studies available. Many employing sensors

2.1 Sensor Research

Sensors are a key element in almost every real-world computing application, especially those concerned with the detection of humans e.g automatic doors. Without them a computing device would have little to no information on its surroundings, and would be extremely ineffective. For this reason, sensors will be a primary component of the smart boxes.

Background research on previously conducted studies gives us a comprehensive list of applicable sensor, and their specific advantages and disadvantages.

2.1.1 Passive Infrared Sensors

Infrared light is the electromagnetic radiation emitted by all objects with a temperature above 0K, giving the feeling of heat. The wavelength of IR light spans from 700 nm to 1 mm, existing just outside of the range of human vision. PIR sensors feature a sensor face that can detect a change in the average amount of IR light hitting the surface, the sensor then outputs a signal indicating this change, with a low voltage indicating low change, and a higher voltage indicating a greater change.

They are most commonly used in PIR-based motion detectors such as automatic lighting systems. Generally the purpose of a PIR sensor is to detect the presence of humans at a binary level i.e true or false. However it has been shown that with suitable positioning and utilisation of machine learning techniques, a PIR sensor can be used to predict the number of occupants in a room within a small range to fair accuracy.

There are many problems to overcome when using a PIR sensor for this purpose, most importantly, the positioning of the sensor; PIR sensors are prone to interference from occluding objects i.e blocking one another from the sensor. As a result of this, the most ideal position would be the ceiling of the room, however this may not prove as practical for other aspects of performance.

Although it will be impossible to obtain a precise estimation by a PIR sensor alone, with the correct implementation, the sensor could be used alongside other components to provide a valid prediction.

2.1.2 Image / Video

The field of computer vision first emerged in the 1960s and has been developing steadily ever since with large abouts of research being done for a number. With endless applications from manufacturing to self-driving vehicles, its not surprising that there is a wide variety of methods available to us, specifically much research has been done on object detection and recognition.

Primitive solutions to object detection might include matching edges from input imagery to predefined templates or a taking and analysing a histogram of oriented gradients (HOG).

In recent years, more complex designs have been produced, often in combination with machine learning algorithms and a training dataset which has been shown to be effective in a variety of object detection problem areas.

Although more complex techniques are tempting, it is important to bear in mind the technical restraints put in place by using a microcontroller; computer vision techniques are notorious for being process intentensive and many require the use of optimised graphical processors to achieve realistic processing times. Fortunately modern digital cameras are extremely advanced and cheap, high quality and compatible digital cameras are readily available, specifically the Raspberry Pi 3 Model B+ features a CSI camera port and a separate camera module. The camera module costs around 20, has an 8-megapixel sensor and is capable of recording in 1080p30 and 720p60 which is most definetely enough for our needs.

To conclude, it seems that computer vision could play a vital role in the final design and is definetely due consideration.

2.1.3 Audio

2.1.4 Other

Chapter 3

Requirements

3.1 Functional Requirements

3.1.1 Sensor data collection

The smart sensor boxes should be capable of collecting data from connected sensors for a specified period of time. The connected sensors we should support are:

1. Bluetooth device scanning.
2. Bluetooth Low Energy device scanning.
3. Camera

However the final system should be easily extendible so that new sensors can be added without too much cost.

3.1.2 Reporting

The smart sensor boxes should be capable of periodically compiling the sensor data into a report and sending it to a centralised storage location.

3.1.3 Estimation

The system should be able to analyse these ”Smart Box” reports, to give an estimation of occupancy for each location with reports.

In order to improve the estimation results, there should be some way of manual readings to be submitted and stored. These readings should then be used in combination with related smart sensor reports to tune the estimation process and producess more accurate results.

3.1.4 Data serving

The report data produced should be made available to authorised users and applications. The estimations produced should be made available to any potential end users and applications.

In addition, authorised users should be able to insert new, delete or amend information on:

- Buildings, floors, rooms.
- Smart boxes.
- Admin users.

3.1.5 Data storage

The system must be capable of storing data about:

1. Buildings, floors, rooms.
   (a) Name.
   (b) Description.
2. Smart boxes.
   (a) Name.
   (b) Description.
   (c) Location.
   (d) Authentication data.
3. Sensor output.
   (a) Time.
   (b) Sensor reading.
4. Smart box reports.
   (a) Time.
   (b) Sensor reading.
5. Estimations.
   (a) Time.
   (b) Location.
   (c) Estimate.
6. Manual readings
   (a) Time frame.
   (b) Location.
   (c) Reading.
7. Admin users.
   (a) Username.
   (b) Hashed password.

3.2 Non-functional Requirements

3.2.1 Sensor data collection

- Ideally each smart sensor box should be as cheap as possible while providing adequte data, in order for the system to be scaled up without large overhead costs.
- For each sensor method, if the data can be collected continuously i.e results can be streamed in real time, then it should be done so with as little downtime as possible. If the data cannot be collected continuously, meaning they require some amount of time to compute results before obtaining a new reading, then we should perform as many iterations as possible to ensure that results are up to date.
- The data collection should be able to run for as long as possible without human intervention to reduce the amount of maintainence required.

3.2.2 Reporting

- To ensure that occupancy reports and predictions are not outdated, RPis should be able to send reports at a rate of atleast one report per minute. The reports should also stick to the specified rate with fair accuracy and not deviate too far from a steady report rate.
- To prevent irregularities in the sensor data affecting reports, the reports should use some form of smoothing mechanism, which would cause any sudden changes in sensor readings to have a smaller effect on the overall estimate.
- Report structure should allow for partial reports from sensor boxes not equipped with all sensors, and also allow for easy extensibility to allow for new sensor additions.
- Report structure should take a common format such as JSON.

3.2.3 Estimation

- The estimates should be as accurate as possible.
- Estimates should be produced at a reasonable enough rate to ensure that they are not out of date.

3.2.4 Data serving

All data served to users should be:

- In a user-friendly and informative format.
- As close to real time as possible.
- Data

3.2.5 Data storage

Any data stored should:

- Be implemented in an efficient and correct manner, without unnecessary dependencies between data.
- Easily scaleable without any major modifications.
- Accessible as quickly as possible and with no down-time.
- Secure from malicious attack or unauthorised access.

Chapter 4

Design

4.1 High Level Summary

We start by splitting the requirements to three seperate components with individual functions.

1. Smart Sensor Boxes: Collects data from the various sensors, also generate reports detailing RPi identification, allong with smoothed sensor data. Reports are then sent to the HTTP server for analysis.
2. Centralised Storage: A database that allows for the storing and querying of data must contain information on building, RPis, estimates, reports, users etc.
3. HTTP Server: A simple HTTP application that can receive sensor report data from the smart sensor boxes. The sever should also be able to perform estimations for locations based on reports and also feature a web application that serves this data to the users.

Figure 4.1 Shows a simple high level diagram of the system design.

4.2 Components

4.2.1 Smart Sensor Boxes

The smart sensor box logic consist of a main reporter algorithm which initialises scanning for each detection method and monitors the output. Every time a report is required, the algorithm takes the outputs from each output queue, and collates them to a report which is sent to the HTTP server for furthur use. Algorithm 3 shows the pseudocode for this algorithm.

The detection methods supported are:

1. Bluetooth / BTLE device scanning - Continuously scans for devices and adds any discoveries to a output map, scans are done for a specified time before restarting to ensure that devices do not stop responding the the scan signal. Algorithm 1 details the algorithms used.

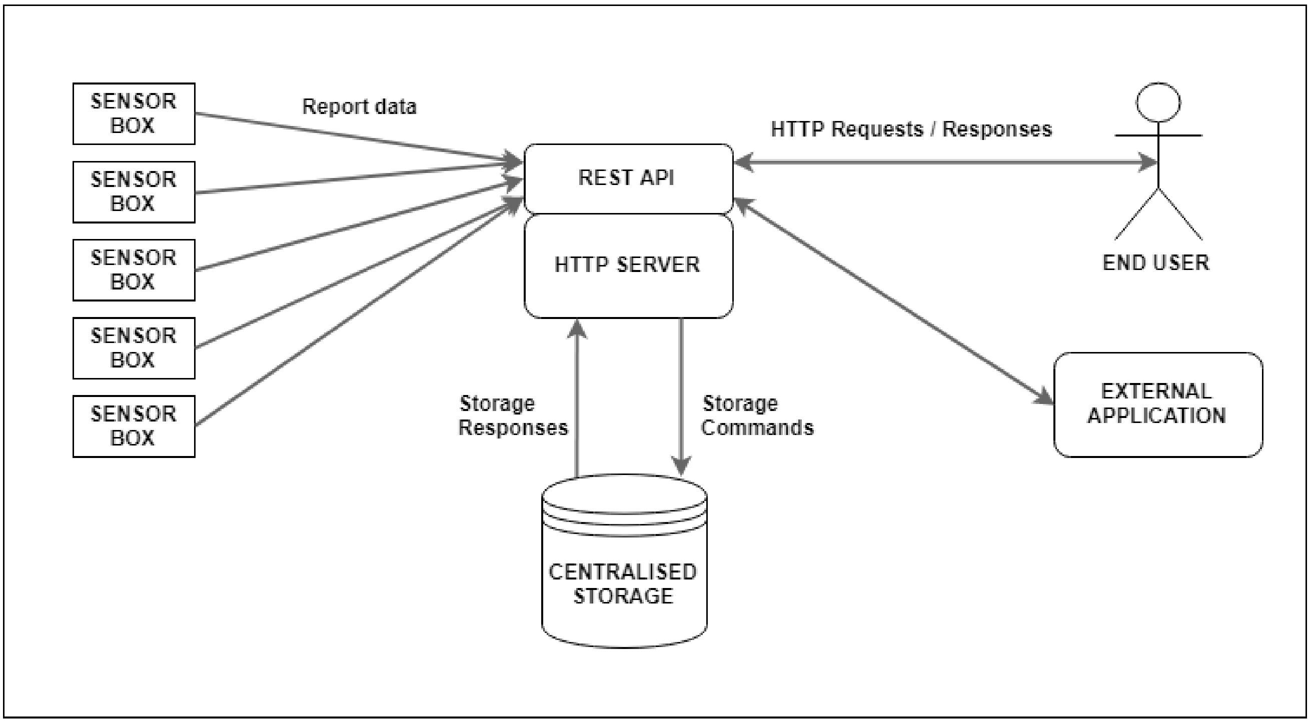

Figure 4.1: Hight level design of system.

2. Camera and object detection - Runs on a loop taking images and feeding them into an object detection algorithm which predicts objects and their certainty. The number of people is extracted from the detection output and smoothed with previous results using an average of the previous value and the new prediction. Algorithm 2 details the algorithms used.

4.2.2 Centralised Storage

- Show ER diagram.
- Discuss relations.
- storing passwords as hash

4.2.3 HTTP Server

```
void scanBluetooth(Int T_cycle, Int T_timeout, Int T_decay, Map output)
begin
    while T_timeout > T_now do
        scan(T_cycle, T_decay)
        while scanning do
            discovery ← new_discovery
            handleDiscovery(T_decay, discovery, output)
        end
    end
end

void handleDiscovery(Int T_decay, Map discovery, Map output)
begin
    addr ← discovery['address']
    rssi ← discovery['rssi']
    time ← discovery['time']
    if address in output then
        old ← output[addr]
        if old['time'] + T_decay < T_now then
            new ← {'rssi' : (rssi + old['rssi'])/2, 'time' : time}
            output[addr] ← new

        else
            new ← {'rssi' : rssi, 'time' : time}
            output[addr] ← new
        end
    else
        new ← {'rssi' : rssi, 'time' : time}
        output[addr] ← new
    end
end
```

Algorithm 1: Pseudocode for Bluetooth and Bluetooth Low Energy device scanners.

```
void scanCamera(Int T_cycle, Int T_timeout, Int output)
begin
    while T_timeout > T_now do
        img ← takeImage()
        objs ← objectDetection(img)
        hum ← numHuman(objs)
        output ← (output + hum)/2
        wait(T_cycle)
    end
end
```

Algorithm 2: Pseudocode for the camera detection algorithm.

```
void reporter(Int T_cycle, Int T_timeout)
begin
    T_decay, T_ccycle, T_bcycle ← getConfig()
    auth ← getAuthData()
    bt_output ← new Map
    cm_output ← 0
    scanBluetooth(T_bcycle, T_timeout, T_decay, bt_output)
    scanCamera(T_ccycle, T_timeout, cm_output)
    while T_timeout > T_now do
        devices ← bt_output
        people ← cm_output
        report ← {'devices' : devices, 'people' : people, 'time' : T_now, 'auth' : auth}
        sendReport(report)
        wait(T_cycle)
    end
end
```

Algorithm 3: Pseudocode for the reporter algorithm.

Chapter 5

Implementation

5.1 High Level Summary

5.2 Components

5.2.1 Smart Sensor Boxes

5.2.2 Centralised Storage

5.2.3 HTTP Server

Chapter 6

Evaluation

Chapter 7

Conclusion

7.1 Summary

7.2 Future Work

7.3 Reflection

Appendices

Appendix A

Title of appendix

Put some data in here:

```
> blahblahblah
```

Bibliography

[1] Test Test. http://www.test/.