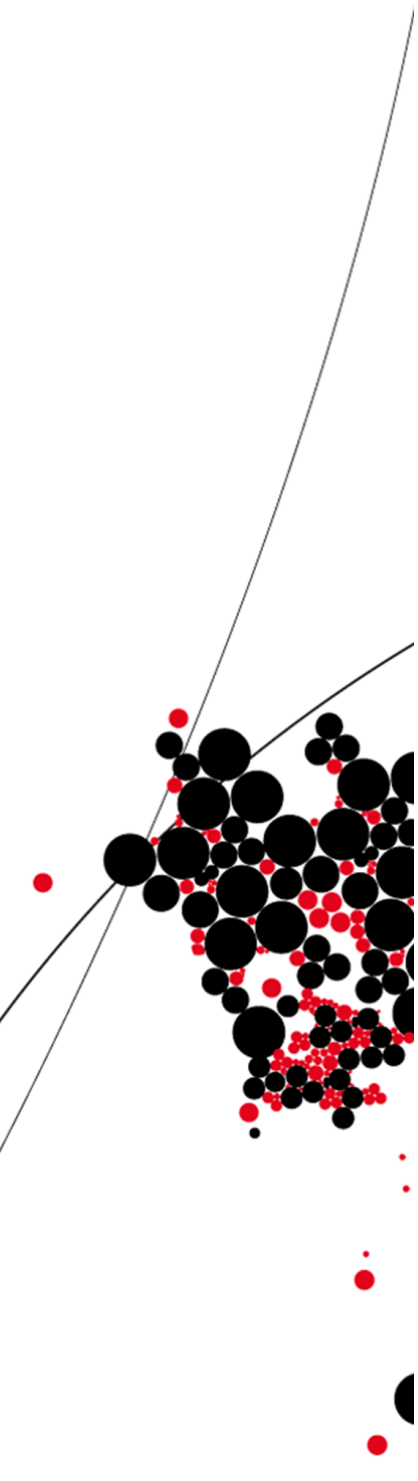
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science



Detection of the crowdedness of a place sensing the devices in the area

Alejandro Ozaita Araico

M.Sc. Thesis
February 2017

Supervisors:
dr. M. Baratchi

Graduation committee:
dr. ir. G.J. Heijenk
prof. dr. M.R. van Steen

Design and Analysis of Communication Systems group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

Nowadays, more and more people start to live in cities. This change involves the apparition of new problems that could be solved using ICTs, which would lead to ”Smart Cities”. In said cities, all kinds of data is gathered thanks to the sensors all around them, and different applications can be developed, such as the detection of crowded places. The detection of these places can be used, for example, for the prevention of human stampedes, for traffic redirection or for reporting the status of a place remotely. The motivation for this research of the detection of crowded places is caused by the small amount of literature that specifies what was considered a crowded place and that most of the existing methods for their detection distinguished between ”crowded” and ”not crowded” areas arbitrarily.

In this thesis, a method for the detection of crowded places calculating the threshold which distinguishes the two mentioned states is presented. For this end, procedures for inferring the number of people, the maximum capacity of an area, and the calculation of the crowdedness threshold using mobility data are described. In conjunction with the description of the methods, their validation in three different areas is also presented.

The results of the validation show that the use of a linear regression model for inferring the number of people in a certain area, is an appropriate approach, as the obtained R-Squared value was acceptable, but its performance could be improved by gathering more ground-truth data for the training phase. Regarding the algorithm for the calculation of the maximum capacity, a possible maximum capacity was calculated for each of the analysed areas, but it was inferior to the considered as ground-truth. Finally, thresholds for detecting crowded situations at each of the areas were calculated.

Contents

Chapter 1

Introduction

Nowadays more and more people are starting to live in the cities. In fact, their population growth rate it is so high that it has been anticipated that by 2050, the 75% of the world population will live in cities [1]. This change involves some problems, like difficulty of waste management, scarcity of resources, air pollution, human health concerns, traffic congestions, optimization of energy and water usage and savings, employment generation, etc.

In a similar way to cities, the ICTs (Information and Communication technologies) have had great developments. Some of the most remarkable ones would be: smartphones, sensors, cloud computing, the semantic web and the IoT (Internet of Things). Due to the improvement of these technologies, there have been several attempts to resolve the aforementioned problems using them.

The incorporation of ICTs to improve and resolve the cities' services, management problems, and other issues, leads us to the "smart cities". A Smart city denotes an interconnected and intelligent city [2]. In order to make the interconnection possible, the development of broadband infrastructure combining several technologies like cable, optical fibre and wireless networks would be necessary, offering high connectivity and bandwidth to the citizens and organization located in the city. It would be also necessary to enrich the physical space and infrastructures with embedded systems, smart devices, sensors and actuators, to be capable of obtaining real-time data and offering real-time services. To be capable of having an "intelligent" city, the creation of applications enabling data collection and processing, would be also necessary [3].

In a Smart city, thanks to data gathering devices and actuators, a large amount of different applications could be developed. Some of these applications could be: the detection and identification of different Points of Interest, targeted advertising, the

optimization of public transport, traffic flow and parking systems, the reduction of CO_2 emissions, for example.

Also, another possible application is to detect if a place is crowded. This system could reveal events in different areas such as parades, festivals or street shows, making possible the prevention of stampedes like the ones occurred in 2014 at Shanghai (China) on New Year's Eve [4] and at Duisburg (Germany) in 2010 [5]. Additionally, this could be implemented to focus on transport flows creating systems that detect traffic jams, accidents or population peaks in public transport such as buses, underground, etc., which could be used to find a pattern and optimize the number and frequency of said means of transport, or to send alerts and, in certain moments, increase their numbers or frequency. In a similar way, the detection of crowded areas could be used to inform the people of the condition of a place remotely, for example, if a person wishes to study, he could check the status of his usual library to evaluate if there may be space for him, and after checking it, the person would be able to consider if the travel to the library is worth it or not.

1.1 Motivation

From the different applications mentioned until now, the detection of crowded areas has been chosen as the research topic for this thesis. This decision has been motivated by the fact that there are different researches focused on the creation of systems thought to perform in crowded areas, but they do not specify what is considered ”crowded” [6] [7] [8].

Due to this fact, an investigation of the existent literature about the detection of crowded spaces was carried out. The findings were that it was limited, and in the existing researches, the threshold to distinguish ”crowded” and ”not crowded” situations were arbitrarily established by the researchers. In consequence, in this project it is desired the creation of a method that detects if a place is crowded or not, calculating the threshold to split those situations using mobility data.

1.2 Challenges

In the chosen research field, there are different challenges that must be overcome. First of all, the conversion of the gathered mobility data to numbers that represent people. This matter can be an issue as, depending on the analysed area, the people

in it may use several devices capable of generating that mobility data, or maybe none, which would discard the possibility of making the equivalence of one data source, one person.

The second challenge to be faced is calculating the maximum capacity of the analysed area, as the crowdedness of it will be affected by the number of people that it is capable of containing. This can be a very difficult matter as this capacity may change depending on the behaviour of the people, which can be hard to predict due to its high variability.

Finally, the last problem that is faced is the creation of a threshold that would divide the ”not crowded” and ”crowded” states. The concept of crowded may vary depending on the person and space, however, as each person’s perception of ”crowded” cannot be known, the creation of this threshold should be based on what, in general, people perceive as ”crowded”. In addition, as it is not desired to depend on people’s reports about the crowdedness of the analysed area, but to calculate the limit using mobility data, the general perception of ”crowded” should be inferred by the observed behaviour of the people that access it.

1.3 Research Questions

From the explanation in the motivation, the main research question is formulated:

- Can we detect if a place is crowded using the mobility data of the devices in the area?

This question implies other three secondary questions, as was implied in the challenges:

- How can we determine the number of people in an area?
- Can we calculate the maximum capacity of an area using the mobility data of the devices in it?
- How can we determine if a place is crowded or not using the mobility data of the devices in it?

1.4 Contributions

The main contributions of this research are: the design of a method for the calculation of a limit to distinguish if a place is crowded or not, not based on arbitrary thresholds but instead on the observed behaviour of the devices. In a similar way, a method is proposed to calculate the maximum capacity of an area considering the existence of data of devices that, even if they are detected, should be discarded as they may not be using the analysed zone. Finally, a method for the conversion of the detected data to people is explained.

1.5 Outline

The following chapter will provide background information on the mobility data and the different ways of collecting it, along with the explanation of the related investigations that try to answer similar research questions to the ones presented in this project. Chapter 3 describes the proposed procedures for answering the research questions. Chapter 4 shows the validation process of the proposed procedures in a testbed and the obtained results. Finally, in Chapter 5, a conclusion of this project is drawn.

Chapter 2

Background and Related work

The first section of this chapter will talk about the mobility data and the different ways to collect its traces. Then, the next section will introduce different researches that use this data to answer the proposed questions in this project.

2.1 Background

Mobility data is described as a set of position records that make possible the calculation of one object's trajectory, if they are chronologically ordered. These traces contain information as basic as the object identifier, the timestamp of the detection and the position of the object when it was detected [9]. There are generally three ways to collect the mobility traces: monitoring locations, monitoring communications and monitoring contacts [10].

2.1.1 Monitoring location

In this category, a wide range of technologies is included. The most widely used technology for outdoor localization systems is the Global Positioning System (GPS), which is based on satellites and provides location accuracy within a few meters. Since this technology requires a line of sight between the monitored device and the satellites, its use is not possible in areas in which there is a high shadowing effect, or indoor areas.

For those cases in which the GPS cannot be used, the location of devices can be determined using the WiFi technology [11]. This approach has been popular in the last few years due to its low deployment cost, potential for reasonable accuracy and

readiness to be applied to mobile devices. The existing WiFi-based solutions usually belong to one of two categories: fingerprint based solutions [12] [13] [14], or model-based approaches [15] [16] [17]. The first solution fingerprints different locations in the area of interest and then searches for the best matching position. On the other hand, the model-based approaches train a signal propagation model using training/calibration data and then trilateration for localization of the objects. These methods have shown to be promising as, under lab conditions, they have achieved below 10 meters' accuracy. Nevertheless, large-scale accurate indoor localization systems have to be developed, as in a real-world context, the localization accuracy of existing approaches in large spaces such as shopping malls and airports can still be up to 20-30 meters.

Finally, if a sub-inch distance precision is required, the RFID technology can be used, in which systems are formed by RFID tags and readers. A disadvantage exists, however, where for being able to detect an object with an RFID tag, the reader has to be extremely close to it. Nevertheless, it is not necessary to be a line of sight between them. Due to its characteristics, the RFID technology has been used in cards for controlling access to different areas and for making electronic payments possible. It has also been used for tracking assets, like robots, in indoor spaces [18].

2.1.2 Monitoring Communication

Another alternative to obtain mobility traces is to use the communication systems, and to monitor the communication of the traced devices, as is the case of this project. The position of a device can be calculated by obtaining the strength of the signal between the base station/access (BS/AP) point and itself. A second method would be checking the connectivity events of the device, since in GSM or WLAN, when a device connects to a cell/access point, it is assumed that it is close to the BS/AP.

This approach is not very precise, as the location data only provides information on whether the mobile device is within the transmission range of some BS/AP. Nevertheless, it can be used indoors as an inexpensive way to locate a node in a specific area (for example, a room) or to validate assumptions of microscopic mobility models. In addition, the accuracy of approximate data can be improved by applying methods of data fusion for tracking [19]. The goal of those methods is to associate, correlate and combine information from a single or multiple sensors to achieve precise estimations.

2.1.3 Monitoring contacts

This approach obtains mobility traces using mobile devices to sniff other mobile devices around them. The traces obtained by this method are called ”contact traces”, since the detected devices are considered ”contacts” and they can be traced using Bluetooth or WLAN in a infrastructureless mode. Also, as the devices may be mobile, the obtained traces cannot be mapped on absolute locations. Nevertheless, for some type of networks, such as opportunistic networks, contacts between the mobile nodes may be more interesting than the localization of the nodes. Contact traces can be used to examine movement and social characteristics which can be used to develop new models and validate existing ones [20].

2.2 Related work

For the development of this project, it was desired to be able to infer the number of people in an area, to know the maximum capacity of it, and to calculate a threshold which would determine if it was crowded or not. In the following sections, researches that try to answer those matters will be explained, followed by a review of their shortcomings.

2.2.1 Detecting people

For the detection of people in a place, it is quite popular to use algorithms based on image or video processing. Said algorithms are based on the recognition of objects' parts that can represent a person (e.g.: faces [21], heads [22] or heads and shoulders [23]–[25]). Nevertheless, these approaches can have several downsides: it is necessary to install cameras for monitoring the desired areas which can have a high cost, and it may be necessary to have an uncomplicated background, etc. Low-cost alternative methods for inferring the number of people in a place can be used as alternative.

For instance, [26] measures the occupancy of a room using a PIR sensor. In their research, a single passive infra-red (PIR) sensor was used, installed in a room which would measure the motion patterns detected at different time windows. Then, the information was used to create a machine learning model to estimate the number of people in the room.

A different method was used by Zhou et al. [27]. In this investigation, the researchers

tried to characterize the educational behaviour of the students of a university measuring parameters such as the attendance ratio or the students' punctuality to the lectures. They used the WLAN of the university to capture the mobility data of the students' mobile devices. However, in order to complete such objectives, they had to solve how to link the number of devices detected in one place to the number of actual people in it. It was not possible to assume that one device corresponded to one person as an account could be used in different devices at the same time. In the research, an app was developed which helped its users to manage their network account on their devices, and also automatically log them into the WLAN. Due to this, once a device was logged in, its MAC address was associated with the used account on that device and the number of logged accounts was used as the number of people in the area.

In addition, in [28] the researchers also used people's smartphones to detect the number of people in an area. However, this approach was focused on the use of the microphones installed in them. For reaching their objective, they use unsupervised learning techniques that allow them to infer the number of people in a conversation or at its surroundings (but not their identity), having all the computation in the smartphone itself. This approach was tested in different environments: quiet (e.g.: homes, offices) and loud (e.g.: restaurants, shopping malls or public squares). As results, the difference between the estimated number of people in the quiet environments and the actual number of people is, on average, slightly over 1, while this error was not larger than 2 in the noisy outdoor environments.

2.2.2 Maximum capacity

The detection of the maximum capacity of an area is, for the best of our knowledge, a not a very investigated area. Nevertheless, in [1], the researchers tried to calculate the moments in which a train station will be crowded. To do so, they collected the number of times that the RFID cards which granted access to the train station were used and the maximum of readings collected in a time window, arbitrarily specified by the researches, as the maximum capacity of the station.

In a similar way, Google's search engine gives a prediction of the crowdedness of a business in comparison with the maximum number of devices that were located on that place. In this case, Google counts the number of devices which have the option "My Location" from Google Maps activated in each of the hours of a day and this maximum is calculated using the data of the past two weeks.

2.2.3 Crowd detection

In the matter of crowd detection, there are two approaches: on one hand is the approach in which an area (halls, concerts, races, parks, etc.) is selected and, inside it, spots which are considered crowded or not, are identified. In this type of researches, the crowdedness of a location is calculated by measuring the ”crowd-density” of them [29], [30]. The ”crowd-density” is the number of people in the analysed scene or the number of people per square meter [31]. For the first case, in [32], the number of people to determine if a place is crowded is arbitrarily established. For the second case, however, it seems to usually be considered that the spot is crowded when there are 2-3 people per square meter, and beyond that limit, it is considered dangerous for the people, as the possibility of accidents on those levels are high [33], [34]. Nevertheless, in [4], instead of using a fixed number for the analysed areas, an algorithm was implemented to calculate the threshold from which a square meter was considered crowded. The number of people per square meter was collected enabling Baidu Map's positioning function in people's smartphones from the analysed places (stadiums or tourist places). On each of the analysed days, the highest amount of people per square meter was chosen and all those values were considered to follow a normal logarithmic distribution. Due to this fact, those numbers are transformed to logarithmic values and their mean (μ) and variance (σ^2) are calculated. Finally, those values are used to calculate the threshold using the following equation:

$$\omega = \hat{\mu}_{peak} + 3\hat{\sigma}_{peak} \tag{2.1}$$

On the other hand, there are other researches that consider, in a more general way, if a place is crowded or if it is not, instead of labelling spots inside them. In these approaches, the criteria to determine whether a zone is ”crowded”, is different in each study. For instance, in the mentioned [1] in which the objective was to estimate whether a train station was crowded or not, to create the threshold that distinguishes those states, the considered maximum capacity of the station was multiplied by λ. This variable λ was manually given different values (0.5, 0.6, and 0.8) in order to consider that the place was crowded depending on the peoples concept of ”crowd”.

In a similar way, in [35] it was tried to estimate the congestion in train cars using the RSSI value of Bluetooth devices. In this case, it was known that each train wagon had a total of 40, 44 or 54 seats (depending on the evaluated train) and it was considered that when 60 people were detected in the same wagon, it was crowded.

There is another approach in which instead of arbitrarily determining a threshold for determining the crowdedness of an area, the people could report their opinion on the crowdedness. This was done using a mobile app in which the users reported the

crowdedness of their public transport by selecting one level out of 5 of crowdedness [36].

2.2.4 Short-comings of the presented methods

As it has been shown, the existing solutions in the literature for answering this project's research questions have different limitations.

In the case of people detection, if a solution based on image or video processing is used, the costs will be high and there may be problems with the positioning of the cameras among others. Nonetheless, the low cost alternatives also present problems such as not being able to properly infer the number of people as they do not manifest themselves (not using the expected WLAN in [26], being static or covered in [27] or remaining silent in the case of [28]).

Regarding the maximum capacity calculation, the presented method presents the problem of not taking into consideration devices that should be filtered out. Due to this fact, in scenarios in which devices that are passing close to the analysed area will be detected, generating faulty results. Also, the obtained maximum it is not confirmed to be the real maximum of the area or simply the highest record of people found in it.

Finally, the methods for crowd detection have different limitations, depending on the type of of the approach used. In the case of crowd-density approaches, they require precise positioning systems to be able to measure the existing density in a small area. This could be a problem in indoor scenarios since there are no widespread accurate positioning solutions. On the other hand, in the other approach, the threshold to determine whether a place is crowded or not, is arbitrarily established.

Chapter 3

Methodology

In this chapter, the general methodology used to complete this project is defined. Firstly, the necessary resources will be described, continuing with a general explanation of the methodology and concluding with the detailed approach which plans to answer the Research Questions.

3.1 Data overview

In order to investigate the crowdedness of an area, mobility data generated by the devices in the zone is used. Each detection is a tuple *(a, s, ts)*, being *a* the hashed MAC address of the detected device, *s* being the id of the sensor that detected *a*, and *ts* being the timestamp of the detection. In addition, to validate the proposed approach, ground truth of the number of people that were in the different analysed areas is also needed. It consists in counting the number of people in each zone during different moments of one day, during several days.

3.2 General procedure

As mentioned in Chapter 1, in this thesis, the main research question to be solved is whether a crowded place can be detected using mobility data gathered from the devices in that area. Hence, three matters were necessary to be accomplished:

- The inference of the number of people in the area using the detected devices in it.
- The calculation of the maximum capacity of the analysed area.

- The estimation of the threshold to differentiate the ”crowded” and ”not crowded” states.

This tasks are solved in the following order:

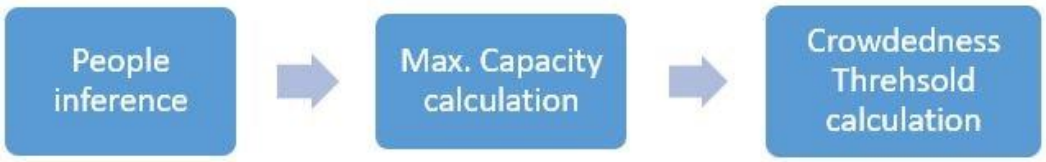


Figure 3.1: General methodology

3.2.1 People inference

In order to know the status of an area during each minute of a day, it is necessary to know the devices that arrive, leave and are currently inside each of the analysed areas. Hence, the first step to be made is the classification of the readings in these groups:

1. New_addresses: this group contains the readings from the devices that have been considered new in the area during any minute of one day.
2. Gone_addresses: this group contains the last readings from the devices that have left the area during any minute of one day.
3. Detected_addresses: it contains the readings from the devices that are inside the selected area during any minute of one day.

Once the number of the devices in each area, during any moment of a day, is known, it is possible to transform the obtained number of MAC addresses into the number of people in the analysed zone (Figure 3.2).

Figure 3.2: People inference methodology

This conversion would be possible using the number of MACs detected in an area and the gathered ground truth of the actual number of people in the area as training data for a regression model.

3.2.2 Maximum capacity calculation

The following task would be the calculation of the maximum capacity of an area. For this end, different information is needed (Figure 3.3).

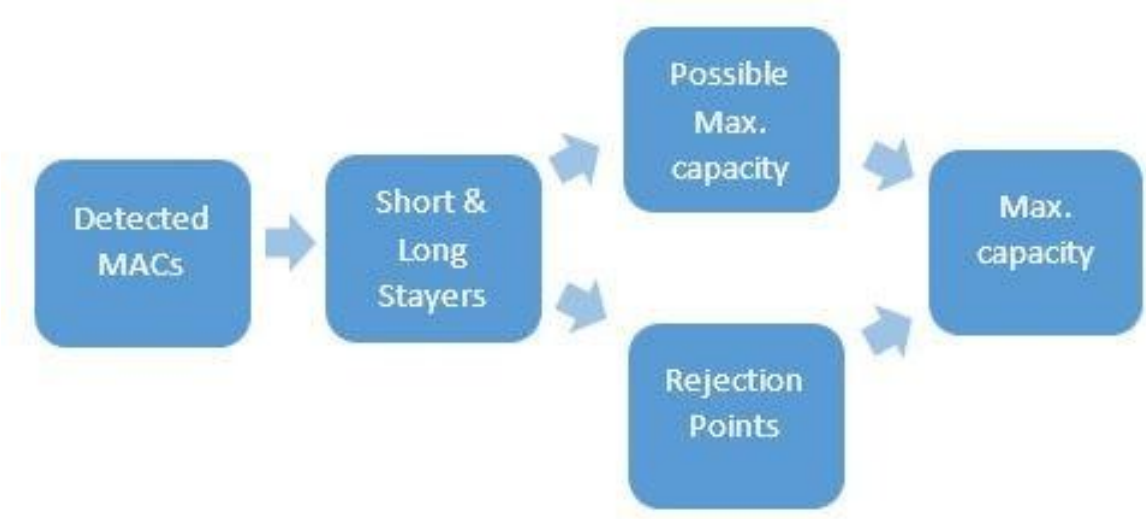


Figure 3.3: General methodology for the calculation of the maximum capacity of an area

First, it is necessary to distinguish between the devices that stay in each of the areas for a small amount of time, and the devices that stay in them for a long period of time at them (”short stayers” and ”long stayers”). This distinction is necessary, as when an area starts to be crowded, there will be people that will not find a spot appropriate for them and will therefore leave the zone quickly, or people passing by could be detected by the sensors, etc., thus, they are not using the available space but they are detected. On the other hand, there will be people that, even if the area is crowded, will find a spot for themselves and they will be truly using the facility, for which they will stay a long time. Hence, the highest peak of the long stayers in an area would be taken as the possible maximum capacity of it. It is called ”possible maximum capacity” as it is not known if it is the actual maximum capacity or the highest number recorded until that moment.

Once the classification of the devices is done, the detection of the rejection points is necessary to confirm that the possible maximum capacity is actually the maximum capacity and to detect the crowded moments. The ”rejection points” would be those moments in which the number of new short stayers in proportion to the new long stayers is unusually large, meaning that the examined area is crowded or even full. The ”new short stayers” would be those which, from the new devices that entered the area, belong to the short stayers group. Similarly, the ”new long stayers” would

be those new arrived devices that belong to the ”long stayers” group.

With the rejection points it is possible to check if the possible maximum is, indeed, the real maximum capacity of an area. For instance, if a possible maximum is found, it could have rejection point, or it may not have one. In case of having a rejection point, as all the seats of the zone will not be occupied, there will be people that will enter and stay in the area making the number of new long stayers at that moment greater than zero. On the contrary, if the maximum capacity is reached, a rejection point should happen, as, if new people arrive, they will not find a place to sit and they will leave. Due to this fact, in this rejection point there will be zero new long stayers.

3.2.3 Crowdedness threshold estimation

Finally, the crowdedness threshold would be estimated from the rejection points. In order to get the threshold, the number of long stayers during those moments in one day will be retrieved and their mean will be calculated, obtaining as a result the threshold for that day. To get a general threshold, the means from all the days will be averaged, obtaining the number of long stayers from which the area is considered generally crowded (Figure 3.4). The reason for using a general threshold is that there may be moments on one day in which there may not be detections and, as consequence, the rejection moment cannot be detected, even if the place is crowded.

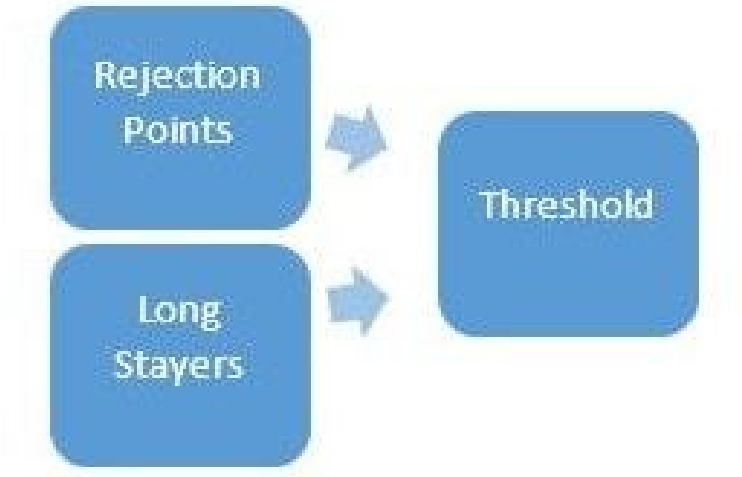


Figure 3.4: General methodology for crowdedness threshold of an area

3.3 Approach

Now that the general approach has been explained, the detailed explanation of the methodology will be carried out. First, the process for the division of the addresses into groups will be explained. Then, the calculation of the criteria for the distinction of the Short and Long stayers is described, followed by the calculation of the Rejection points. Finally, the processes for the maximum capacity and the crowdedness threshold are defined.

3.3.1 Group division

As it was previously mentioned, it is necessary to know the addresses that enter, leave and are inside one area in order to know its status during any minute of one day. However, it is necessary to distinguish between the readings from the devices that are new in the area or that were there,then left and came back, and those readings from devices that were sensed before, did not leave the area, and were detected again. To make this possible, the threshold to consider that a device left the area or that it is still in it, even if it was not sensed, must be obtained. It will be calculated for each day to keep it updated on each analysed day. This time limit is necessary as all the devices in the area are not detected at the same moments and, the interval of times in which one device is noticed is not constant. Finally, the readings should be classified into each of the groups mentioned before (New_addresses, Gone_addresses and Detected_addresses).

3.3.1.1 Devices' time limit

As it was just mentioned, since the devices in the analysed area are not detected at the same times and the intervals in which they are noticed vary, it is necessary to establish a ”Time limit”. It determines when a reading comes from a device that was not in the area and just entered it and when a reading belongs to a device that was in it before and it is sensed again. If the threshold is not used and, instead, the first reading of a device is marked as its arrival and the last reading as its departure, it could have happened that the device leaves the area and comes back several times and it would not be taken into account. Using this margin would make detecting the aforementioned entries and exit possible, and it would also allow to ignore those moments in which the device was not sensed due to interferences or because it was not emitting a signal at that moment.

As every device is sensed at different times and at different time intervals, the mean of those times is used. In order to do so, first all the different MAC addresses detected during the day will be paired with their different timestamps. Then, the difference between the timestamps of each MAC will be calculated and the obtained numbers used to obtain the average time that each MAC is sensed. Finally, those average times are used to obtain one final mean which represents the average time that passes before a device is sensed again. Also, those addresses which only have 1 reading are not taken into account, as those are devices passing by and they will only lower the mean.

This last calculated value is used as the threshold for the day to distinguish between the devices that have just entered the area, the ones that have left the area, and those that were already in the area.

The following pseudo-code shows the described algorithm:

Algorithm 1 Calculation of one day's time limit

```
Input: all the readings from the database
Output: time limit to determine if an address left the area.

address_timestamp_list = []
address_interval_mean_list = []

address_timestamp_list = pair_addresses_timestamps(reading_list) #gene-
    #rates a list of dictionaries which relates each MAC with a list of
    #its timestamps
address_interval_mean_list =
    make_macs_timestamp_interval_means(addresses_list) #generates a
    #list of dictionaries which relates each MAC with the average time that
    #has to pass to be detected
final_mean_seconds =
    make_final_mean_from_means(address_interval_mean_list)
final_mean_minutes = round_to_closest_int (final_mean_seconds/60)

return final_mean_minutes
```

3.3.1.2 Group classification process

Once the time limit is calculated, we proceed to classify the gathered readings of an area to know the number of devices that arrive (”New_addresses”), leave (”Gone_addresses”) and are inside it (”Detected_addresses”), during each minute of a day. In this process, there are a total of 6 groups: ”New_addresses”, ”Gone_addresses”, ”Detected_addresses”, ”Ordered_readings”, ”Addresses_in_this_minute”, ”Addresses_in_previous_minutes”. The important groups are the first three which will have a total of 1440 positions (1 position for every minute in 1 day). The other 3 groups will be used as support to obtain the final classification and the ”Ordered_readings” will also have 1440 positions, while the remaining groups’ longitude is not limited.

Now the classification is explained in detail. It is done in 4 steps:

- Step 0: all the readings extracted from one day from the database are ordered by their timestamp and saved on the ”Ordered_readings” list. From the timestamps, the hour and the minute of the analysed reading will be retrieved and it will be stored in its corresponding position (e.g.: if the time is 1:05, the reading will be stored on the 64th position). Each position will contain a list of readings.

- Step 1: the ”Ordered_readings” list is traversed. As mentioned previously, at each position, there is a list of readings of the detected devices during each minute. When one minute’s reading is processed, its address is searched in the group of
  Addresses_in_previous_minutes”. If it is in the ”Addresses_in_previous_minutes” group, the reading of that address in that group is deleted, in order to not keep addresses with old timestamps.

  Afterwards, the addresses of the readings are searched in the ”Addresses_in_this_minute” group. Those readings whose address were not in the group are added, otherwise, they are not. This is done in order to prevent an address from being counted twice in the same minute.

  Those readings whose addresses were not in previous minutes nor in this minute, are the ones that belong to the devices that just entered the area. Due to it, they will be added to the ”New_addresses” group in their correspondent position, which is marked again by their timestamp.

- Step 2: Once all the readings of one minute of the day are processed, the readings of the ”Addresses_in_previous_minutes” are checked to detect those which have ”timed out”. A reading is considered to have ”timed out” when the

difference between its timestamp and the time of the day that was processed is equal or greater to the time limit calculated for this day (Subsection 3.3.1.1).

If a reading has timed out, the device that created the reading is considered out of the area. Thus, this reading is included in ”Gone_addresses”, once again in the position that correspond to the time of the day given by its timestamp. Those readings that did not time out are considered in the area even if they were not sensed. Consequently, they are added to the ”Addresses_in_this_minute” vector.

Finally, before processing the next minute, the readings in the ”Addresses_in_this_minute” are added to the ”Detected_addresses” matrix in the positions given by the minute that was just analysed. Then, the readings in ”Addresses_in_this_minute” are used to overwrite the ”Addresses_in_previous_minutes” list and then the group is emptied, as the processed minute is changed and new addresses are going to be processed.

- Step 3: the previous 2 steps are done for all the minutes of one day and once all these minutes are processed, a correction must be made. As some devices are considered to be in the area even if they were not sensed, by the time a time out of a reading is detected, its address will have been added to ”Detected_addresses” group several times. In consequence, the ”Gone_addresses” list is traversed, and at the moment that an address is found, the positions that correspond to the next minute until the time limit are taken from the ”Detected_addresses” matrix. From those positions the readings that contain the address found at ”Gone_address” will be deleted.

The classification process can be seen on the Algorithm 2. In order to have a better comprehension of this algorithm, an example has been added to Appendix A.

3.3.2 Short and long stayers’ distinction criteria

In order to obtain the maximum capacity, it would be necessary to distinguish the devices that represent people that were detected but are not using the analysed space (people passing by, people that went there to retrieve something, etc.), from the devices of those people that are using the area. These two different groups will be referred as ”short stayers” and ”long stayers”. The groups can be used for detecting the rejection points and the possible maximum capacity. These can be made obtaining the ratio of the number of short stayers per long stayers in each

Algorithm 2 Group classification process

```
Input: all the readings from the database, time_limit
Output: new_addresses, gone_addresses, detected_addresses

new_addresses, gone_addresses, detected_addresses = []
ordered_readings, addresses_in_this_minute=[]
addresses_in_previous_minutes =[]

#Step 0
ordered_readings = order_readings_by_ts(database_readings)

#Step 1
for(minute=0; minute<len(ordered_readings); minute++)
    reading_list = ordered_readings[minute]
```

rejection point and taking the highest number of long stayers as a possible maximum capacity respectively.

To be able to differentiate between these two groups, it is necessary to know the staying times of the people on the days which do not belong to the high activity period and the ones that do. On the low activity days, there will be more people that stay only a few minutes in the area than on the high activity days, as there is not a great necessity to stay there. However, there will be a moment in time in which there will be a tendency of there being more devices on the high activity days than in the low ones, due to the people's necessity to stay long times to do their activities in that area. This increment on the number of devices will not happen only once, but repeatedly over time, as not every person will stay the same amount of time but most of them will need long times. In consequence, the moment in which the number of devices at the high activity days is higher than in the low ones is taken as threshold to distinguish the "low stayers" and the "long stayers". In order to make possible the comparison between these days, as on each of them different number of readings were collected, the number of readings in each day is normalized.

As seen in Algorithm 3, each day is processed and the devices that were detected on each minute are grouped by the number of minutes that stayed in that area (stay time). Each group contains the number of devices that stay there between an interval of time ("minutes_per_group"). For example, if each group contains intervals of 5 minutes, then, there will be the following resulting groups: Group 0 (devices that stay between 0 and 4 minutes), Group 1 (devices that stay between 5 and 9 minutes), Group 2 (devices that stay between 10 and 14 minutes), etc. Then, the number

```
    for(idx=0; idx<len(reading_list); idx++)
        reading = reading_list[idx]
        previously_detected = is_in_previous_minutes(reading,
            addresses_in_previous_minutes)
        in_this_min = is_in_this_minute(reading,
            addresses_in_this_minute)

        if previously_detected:
            addresses_in_previous_minutes.delete(reading)
        if not in_this_min:
            addresses_in_this_minute.append(reading)
        if not presviously_detected and not in_this_min:
            new_addresses[minute].append(reading)

        #Step 2
        for(idx=0; idx<len(addresses_in_previous_minutes); idx++)
            reading = addresses_in_previous_minutes[idx]
            reading_minute = reading.get_timestamp()
            if (minute - reading_minute < time_limit):
                addresses_this_minute.append(reading)
            else:
                #Time out
                gone_addresses[minute].append(reading)
        detected_addresses[minute] = addresses_in_this_minute
        addresses_in_previous_minutes = addresses_in_this_minute
        addresses_in_this_minute.empty_list()

#Step 3
for(minute=0; minute<len(gone_addresses);minute++)
    gone_array_addrs = gone_addresses[minute]
    for(idx=0; idx<len(gone_array_addrs); idx++)
        gone_addr = gone_array_addrs[idx]
       #Offset starts at 1 to erase the entries after the address was
       #gone
       for(offset=1; offset<=time_limit; offset++)
           examined_gone_array_addrs = gone_addresses[minute+offset]
           position = find_addr_position_in_addrs_list(gone_addr,
               examined_gone_array_addrs)
           examined_gone_array_addrs[position].delete(gone_addr)
return new_addresses, gone_addresses, detected_addresses
```

of devices detected in each group is normalized using the total number of devices detected on that day. Afterwards, the normalized number of devices at each minute of that day is appended to each group on the list of the ”high_activity_days_group_list”, if the analysed day was a high activity day, or to the ”low_activity_days_group_list” if it was not. Finally, the two lists are graphed and the tendency is searched.

Algorithm 3 Tendency detection process

```
Input: day_list #list of the analysed days with the detected addresses
   minutes_per_group #range of stay minutes that each group contains
Output: graph with the medians of the high and low activity days

high_activity_days_group_list = []
low_activity_days_group_list = []
#The length of the two list above is equal to 1440/minutes_per_group
#being 1440 the total number of minutes in a day. Each position is a
#group and each contains a list of numbers which are the normalized
#number of devices

for(a=0; a<len(day_list); a++)
    tmp_group_list = [] #It contains the detected devices on
    #a day, grouped  by the stay times
    normalized_temp_group_list = [] #It contains the normalized
    #of the detected devices on day, grouped  by the stay times

    day = day_list[a]
    total_device_num = day.get_total_device_num()
    detected_addrs = day.get_detected_addrs()

    for(minute=0; minute<len(detected_addrs); minute++)
       addr_list = detected_addrs[minute]
       for(idx=0; idx<len(addr_list); idx++)
           addr=addr_list[idx]
           stay_time = addr.get_stay_time()
           position = int(stay_time/minutes_per_group)
           tmp_group_list[position].append(addr)
```

```
        for(minute=0; minute<len(tmp_group_list); minute++)
            normalized_num = len(tmp_group_list[minute])/total_device_num
            normalized_temp_group_list[minute].append(normalized_num)

        if(day.is_high_activity_day()):
            high_activity_days_group_list.append_for_each_group(
                normalized_temp_group_list)
        else:
            low_activity_days_group_list.append_for_each_group(
                normalized_temp_group_list)

generate_median_graphs(high_activity_days_group_list,
    low_activity_days_group_list)
```

3.3.3 Rejection ratio

Once short and long stayers can be obtained, all the analysed days are processed again. On this process, first, the low activity days are analysed to calculate the ”busy threshold”. The busy threshold represents the number of long stayers on an area from which there are more people than usual and, thus, making crowded moments possible. This division is necessary to limit the analysis to those moments in which rejection points could happen.

The busy threshold is calculated by taking the highest peaks of the low activity days and averaging them, as shown on the Algorithm 4. The low activity days are used as they are not supposed to be crowded at any of the moments and, thus, the crowded moments will be placed over the highest point of the low activity days. However, as exceptions may occur, in order to compensate them, the mean of the highest moments is used.

Afterwards, the high activity days are processed to locate the interval of minutes that may have a rejection point. To do so, the intervals that contain the same number of long stayers or more than in the busy threshold are located (Algorithm 5). The intervals can have a size where, when dividing the total number of minutes in a day, the result is an integer number (e.g.: 1, 2, 3).

Then, all the analysed days are processed to get, from the intervals that were considered busy, the number of short stayers that entered that area at each day at each interval of time (Algorithm 6). The number will be normalized in order to be able to compare the number of devices on each of the day types (high and low activity). The comparison is made using the medians of each of the groups. The medians of the

Algorithm 4 Busy threshold calculation process

```
Input: low_day_list #list of the low activity days with the long stayer
#devices.
Output: busy_threshold

highest_peaks_list = []

for(a=0; a<len(low_day_list); a++)
    day = low_day_list[a]
    long_stayers_list = day.get_long_stayers_per_minute()
    #long_stayers_list contains 1440 positions with the number
    #of long stayers at each position
    max_long_stayers_at_1_minute = long_stayers_list.get_maximum_number()
    highest_peaks_list.append(max_long_stayers_at_1_minute)

busy_threshold = mean(highest_peaks_list)
return busy_threshold
```

Algorithm 5 Busy intervals detection process

```
Input: busy_threshold, interval_size,
high_day_list #list of the high activity days with the long stayer
#devices
Output: busy_interval_list

busy_interval_list = []
for(a=0; a<len(high_day_list); a++)
    day = high_day_list[a]
    long_stayers_list = day.get_long_stayers_per_interval(interval_size)
    #long_stayers_list contains (1440/interval_size) positions with the
    #number of long stayers at each position

    for(interval=0; interval<len(long_stayers_list); interval++)
       if (long_stayers_list[interval] >= busy_threshold):
          busy_interval.append(interval)
return busy_interval
```

Algorithm 6 Rejection intervals detection process

```
Input: day_list,interval_size
Output: rejection_interval_list

for(a=0; a<len(day_list); a++)
    day = day_list[a]
    total_device_num = day.get_total_device_num()
    new_short_stayers_list = day.get_new_short_stayers_per_interval(
    interval_size)
    #new_short_stayers_list contains (1440/interval_size) positions with
    #the number of short stayers that entered the area at each position

    for(interval=0; interval<len(new_short_stayers_list); interval++)
        if(interval in busy_interval_list):
            normalized_value =
                new_short_stayers_list[interval]/total_dev_num
            normalized_new_short_stayers_list.append(normalized_value)
            if(day.is_high_activity_day()):
               high_short_stayers_list.append(normalized_value)
            else:
               low_short_stayers_list.append(normalized_value)

for(interval=0; interval<len(high_short_stayers_list); interval++)
    high_median = high_short_stayers_list[interval].median()
    low_median = low_short_stayers_list[interval].median()
    if( high_median > low_median):
        rejection_interval = busy_interval_list[interval]
        rejection_interval_list.append(rejection_interval)
return rejection_interval_list
```

new short stayers in an area are compared as it is expected that usually on the days in which there are no hurries to finish tasks (low activity days) there will be more people that are passing by the area than in the days that are of high activity, since on those days people will be staying in the area for long periods of time to finish the tasks. Therefore, if there is an interval in which the number of new short stayers of the high activity days is higher than the low ones, it would mean that something is keeping the people from staying in the area. This is taken as a ”Rejection interval”.

Next, from the intervals in which it was considered that people were being rejected, the ratio of the number of new short stayers that are detected in the area is calculated, per each new long stayer (Algorithm 7). In consequence, the feature that can identify a rejection point is known. Finally, the ratio of each interval is averaged in order to know the average ratio on the rejection points and to be able to detect one when processing a day (Algorithm 7).

Algorithm 7 Rejection ratio calculation

```
Input: rejection_interval_list,
new_short_stayers_list #list with the raw number of the new short
#stayers in an area per interval
new_long_stayers_list #list with the raw number of the new long stayers
#in an area per interval
Output: general_ratio

ratio_list = []
for(interval=0; interval<len(rejection_interval_list); interval++)
    ratio =
        new_short_stayers_list[interval]/new_long_stayers_list[interval]
general_ratio = mean(ratio_list)
return general_ratio
```

3.3.4 MAC to people conversion

Once that the number of devices on each area is known, the detected MACs and the gathered ground truth of the actual people in one area can be used to know the number of people in the area from the gathered readings. As the number of readings will be linearly correlated with the number of people, since the number of devices in one area will depend on the number of people in it, a linear regression algorithm can be used to make this conversion.

For this end, the number of people during one moment is paired with its corresponding number of readings. Finally, those pairs are used for the training of the linear regression model, and once this phase is done, it will be possible to predict the number of people from the number of readings.

3.3.5 Maximum capacity

Now that the people that are using the areas to study can be detected, the possible maximum capacity can be obtained (Algorithm 8). First, the long stayers of the high activity days are identified and the highest number of those long stayers is taken as a possible maximum. Then, it is corroborated that the possible maximum is actually the maximum capacity. To do so, we check whether there is a rejection point at the moment in which it was found. When the maximum capacity is reached in an area, any person that tries to stay in it will not have a spot and will leave the zone, because of this, a rejection point should happen at that moment and, also, the number of new long stayers at that moment is zero.

Algorithm 8 Calculation of the Maximum capacity

```
Input: high_activity_days, long_stay_criteria, rejection_ratio
Output: maximum capacity or possible maximum capacity

is_max_capacity = False
is_rejection_point = False
num_long_stayers = -1

possible_max =
    get_possible_max_capacity(high_activity_days,long_stay_criteria)
is_rejection_point, num_long_stayers =
    rejection_point_at_possible_max(possible_max,rejection_ratio)

if (rejection_point) and (num_long_stayers == 0):
    is_max_capacity = True
return possible_max, is_max_capacity
```

3.3.6 Crowdedness threshold

In order to calculate the number of people for which the area is generally considered crowded, from the high activity days, the number of long stayers on the detected

rejection points on one day are used (Algorithm 9). On each day, the long stayers on the rejection points are averaged and the resulting number is the crowdedness threshold for each day. Finally, to get the general threshold, each day's thresholds are averaged. A general threshold is calculated because there may be moments in one day in which rejection points could not happen as there may not be signals passing by and, as consequence, the rejection moment cannot be detected, even if the place is crowded.

Algorithm 9 Calculation of the Crowdedness threshold

```
Input: high_activity_days, long_stay_criteria, rejection_ratio
Output: crowdedness_threshold

threshold_list = []
crowdedness_threshold = 0

for(i=0;i<len(high_activity_days);i++)
    day = high_activity_days[i]
    day_threshold =
        get_crowded_threshold(day, long_stay_criteria, rejection_ratio)
    add_threshold(day_threshold, threshold_list)
crowdedness_threshold = mean(threshold_list)

return crowdedness_threshold
```

Chapter 4

Validation

In this chapter, the previously described methodology will be tested to answer the Research Questions. Firstly, the testbed used for this validation is introduced, along with the used methods to gather the mobility data. In addition to this, the analysed day span and the obtained ground truth are also described. Next, the assumptions that were used are explained, and, finally, the processes used to answer the research questions are described.

4.1 Data acquisition

On this thesis, the University of Twente Vrijhof building's library has been used as the testbed, which contains a total of 3 Wi-Fi sensors. The sensors IDs are 1992, 1994 and 2001, but for a better understanding, they will be called Sensor 1, Sensor 2 and Sensor 3 from this moment onwards. Their function is to capture the mobility data generated by the devices in their coverage area in order to investigate its crowdedness. The position of the sensors can be seen on the maps pictured in Appendix B.

Each detection is a tuple, like the one presented in the methodology, but it has an additional parameter *o*, which contains the OUI of the detected device. Due to this fact, the structure of the tuple in this test is *(a, s, ts, o)*. This additional field is used because, in the analysed areas, there may be devices with randomized MAC addresses and they can be removed by processing them.

About the analysed days, they start on the 5th of September of 2016 and end on the 2nd of February of 2017. From that range, the weekends and the Christmas Holidays were discarded as the people's behaviour on those days differ from regular

working days, and the number of devices on those days is greatly reduced. It is also worth noting that there were some days in which a sensor was not working or stopped working early in the day ("Faulty days"), and those are also discarded. From the remaining days, the days of the high activity period were manually chosen using the official calendar of the University of Twente's official exam days as ground truth. Those days that showed higher activity than normal days in the previous 2 weeks to each of the exam periods and in the exam period were chosen as "high activity days". Finally, those days that did not belong to the "faulty days" or the "high activity days" are the ones belonging to the "low activity days". These leaves the following number of days in each group of days for each sensor:

	Sensor 1	**Sensor 2**	**Sensor 3**
Low activity days	53	77	78
High activity days	23	21	20
Faulty days	22	0	0
Total days	98	98	98

Table 4.1: Used days

In addition, the ground truth of the number of people that were in the different sensors' areas was gathered. It consists on the counting of the number of people in each area covered by the sensors. The counts were manually made from the 18th of October of 2016 to the 25th of November of 2016 at five different hours (10:00, 11:30, 14:00, 19:00 and 21:00). Also, the number of total seats in each area was counted.

4.2 Assumptions

In order to answer the research questions, it was assumed that the analysed areas would have high activity periods. In this case, as the sensors are placed in a university library, it would be the exam periods and their previous weeks, as the students would use the facilities to finish their assignments and to prepare themselves for the upcoming exams. In these high activity periods, there is believed to be a higher probability to have the chosen areas at their maximum capacity than outside of the exam period, where less students would feel the necessity to go to the library.

To answer the question of how to obtain the number of people in the area, a very simple assumption was made: the number of detected MACs will be related to the number of people in the area.

4.3 Validation process

The following subsections contain the application of the proposed methodology in the context of this project.

4.3.1 Group classification

The first step to be made is the calculation of the Time limit to detect if the analysed reading belongs to a device that just arrived, that left and came back, or simply to a device that was sensed before and did not leave, and then the readings are classified into the previously mentioned groups (new, gone and detected in the area) will be made. However, as in this case the analysed devices can have randomized MAC addresses, they must be detected and then removed, in order to not sense one device and count it as 2,3,4, or more different devices. To make the detection possible, the OUI field of the analysed MAC is used and this is done before anything else.

The MAC addresses can be either universally administered or locally administered addresses.

- The universally administered addresses are uniquely assigned to a device by its manufacturer. They have a total of 6 octets, the first 3 identify the organization that gave the device the identifier (these octets are known as OUI), and the remaining 3 octets are assigned by the organizations in any manner they please.
- The locally administered addresses are assigned to the devices by a network administrator and they substitute the original address provided by the manufacturer.

The two types of addresses are distinguished by setting the second least significant bit of the OUI part of the address. If the bit is 0, the address is universally administered, and if it is 1, it is locally administered (Figure 4.1). As the randomized addresses will be a type of locally administered address, in order to filter them out, those OUIs whose second least significant bit is set to 1, will be filtered out. However, doing this may introduce an error as not all the OUIs with the bit set to 1 implies that the address is randomized, but as those addresses are exceptions, the error is considered acceptable.

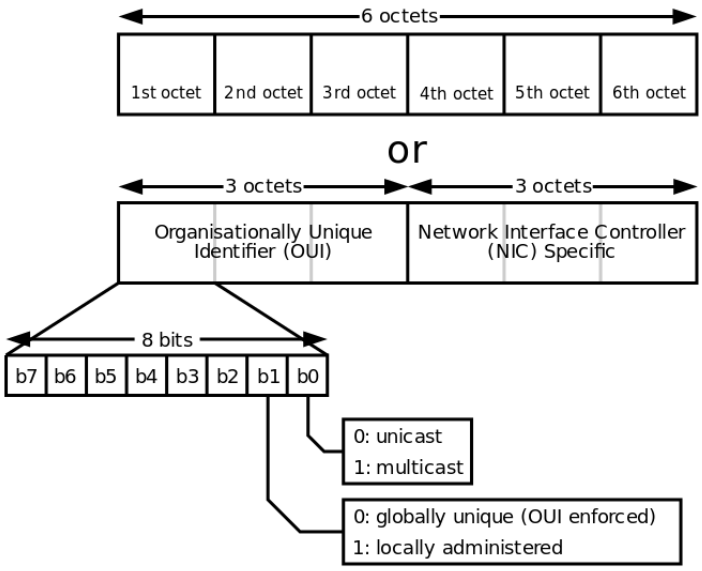


Figure 4.1: MAC address structure

The reading filtering process is shown on Algorithm 10. First, the analysed readings OUI attribute will be retrieved and, as it will be an hexadecimal value, it will be transformed to binary. Then if the value of the second least significant bit is 1, the reading will be discarded, and if it is not, it will be used for the rest of the process.

Algorithm 10 Detection of a Randomized MAC

```
Input: one reading from the database
Output: True, the address is probably randomized, or False, it is not.

oui_hex = reading.get_OUI()
oui_bin = transform_to_binary(oui_hex)
if get_second_least_significant_bit(oui_bin) == 1
    return True
else:
    return False
```

Once the devices with randomized MACs are removed, the time limit is calculated and the readings are grouped for one day. It is possible to observe the number of people that were in one area during a certain day (Figure 4.2).

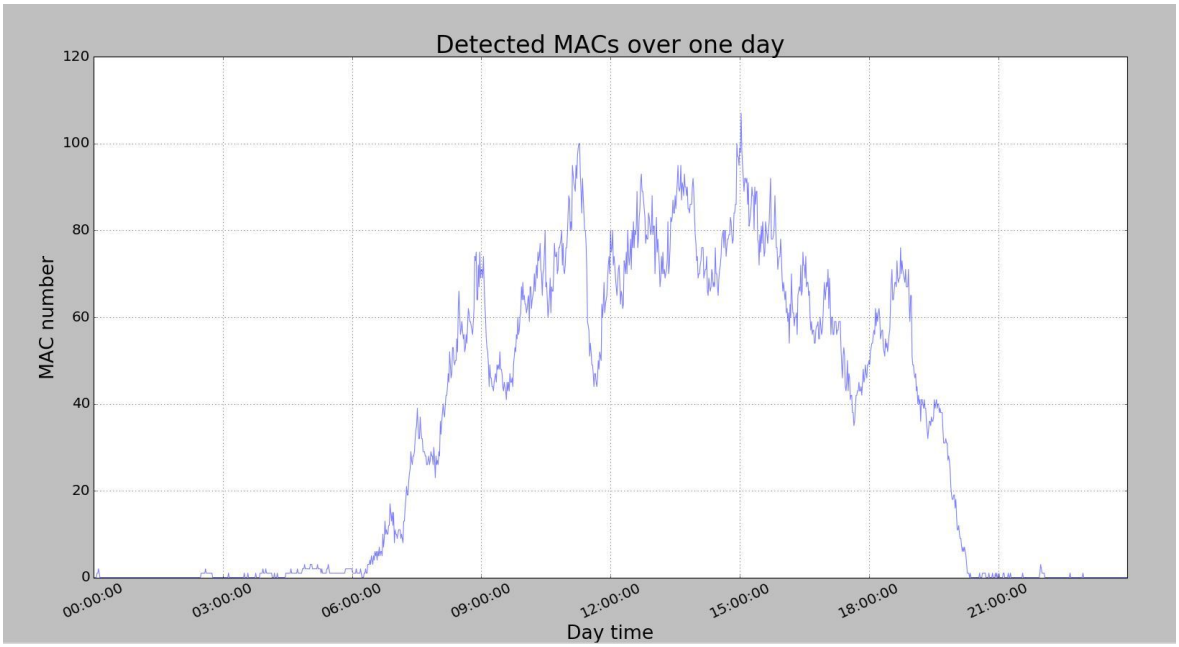


Figure 4.2: Detected MACs in one day example

4.3.2 Short and long stay criteria

After obtaining the capability of sensing the different devices in an area, it is calculated the threshold between the short and long stayers to later detect rejection points and get the general ratio for their detection. As previously explained, there are two types of days: low activity days and high activity days. In the first type, as there is not a great necessity to stay in the area, there will be a great number of devices that will have low stay times and only some with high stay times. On the other hand, on the high activity days, there will be only some devices that will stay a few minutes while a great number of the devices will have high stay times. Hence, it is expected that showing the number of the devices on each possible minute of stay time, on the first minutes, the number of the devices that belong to the low activity days will be higher than the number of devices of the high activity days. However, as the stay time increases, there will be a point where the number of devices from the ”high activity” group will be greater than those of the ”low activity” group, and this will be the case for a long time. The minute in which this tendency of having more devices on the low activity days than on the high activity days is reversed, it is considered as the threshold that a device has to pass to be considered a long stayer.

For the calculation of the threshold of the short and long stayers, firstly, the devices that were detected during each day are retrieved with their stays times and they are grouped by them. However, before grouping them, the number of devices that were detected during each day is normalized in order to be able to make the comparisons.

Then, these groups are made every minute of stay time, which means that in Group 0 there will be the devices which have stay time of 0 minutes (between 0 and 59 seconds), in Group 1 those which have a stay time of 1 minute (between 60 and 119 seconds), etc. Groups representing each minute of stay time are used, as it was desired to have clusters representing small fragments of time to be able to detect the threshold between short and long stayers accurately. Inside of each group, another two clusters are created splitting the devices that were on the high and low activity days. Finally, the medians of the groups are used to locate the tendency change(Figures 4.3,4.4 and 4.5).

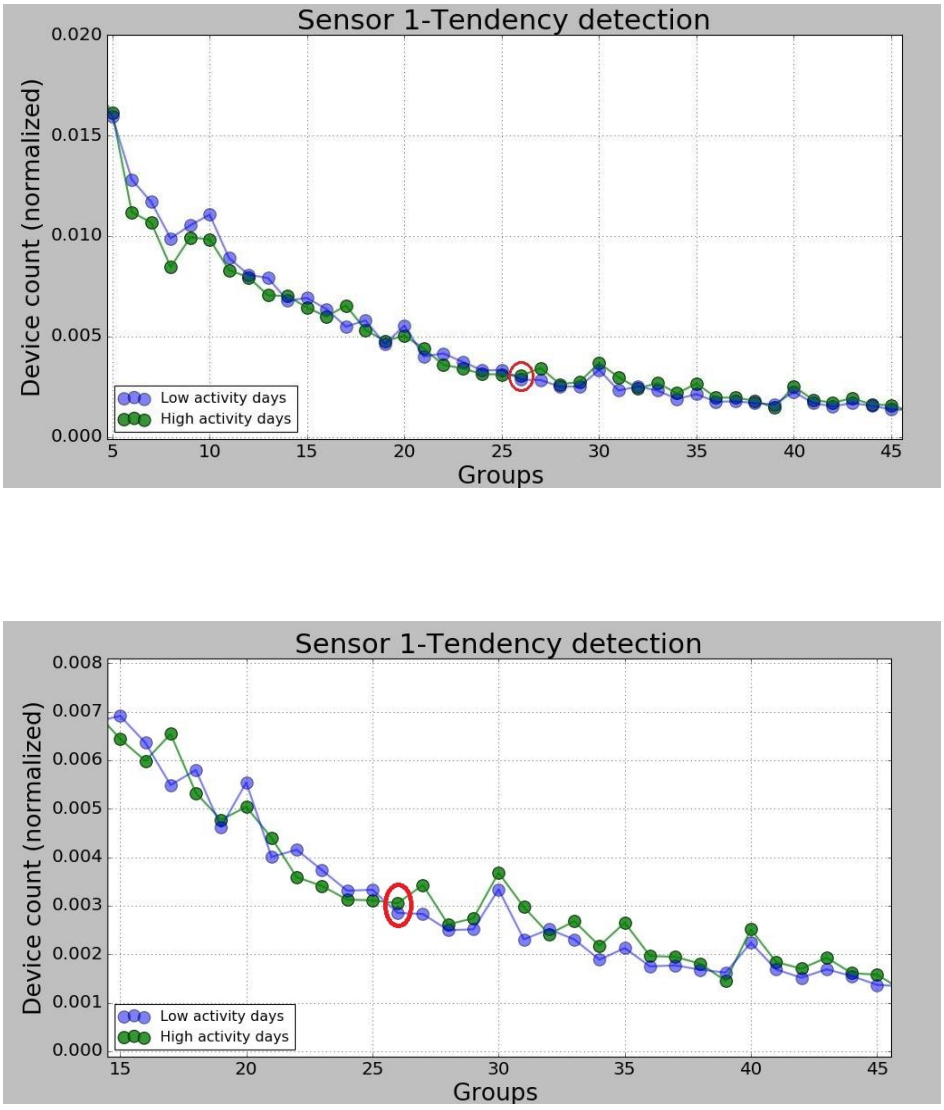


Figure 4.3: On sensor 1, there are more devices on low activity days until the Group 26, from which the tendency is reversed and then, there are more devices on high activity days. General view above, zoomed in view below

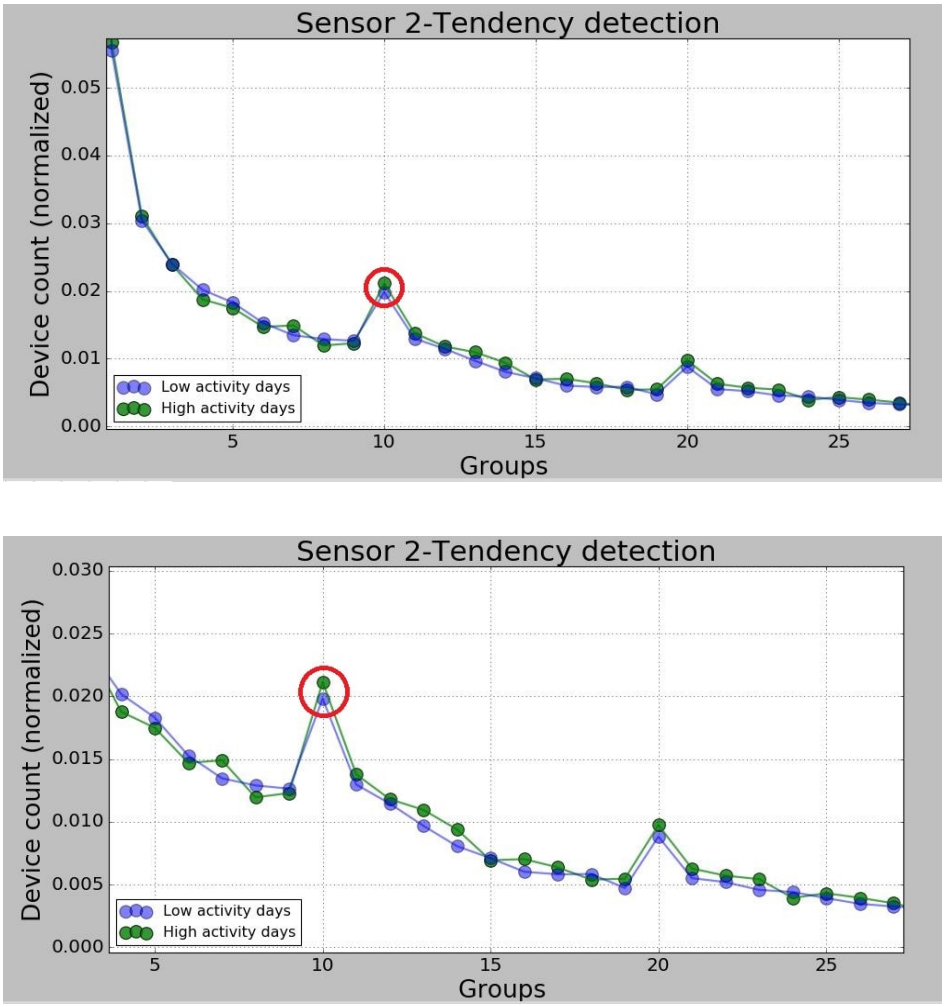
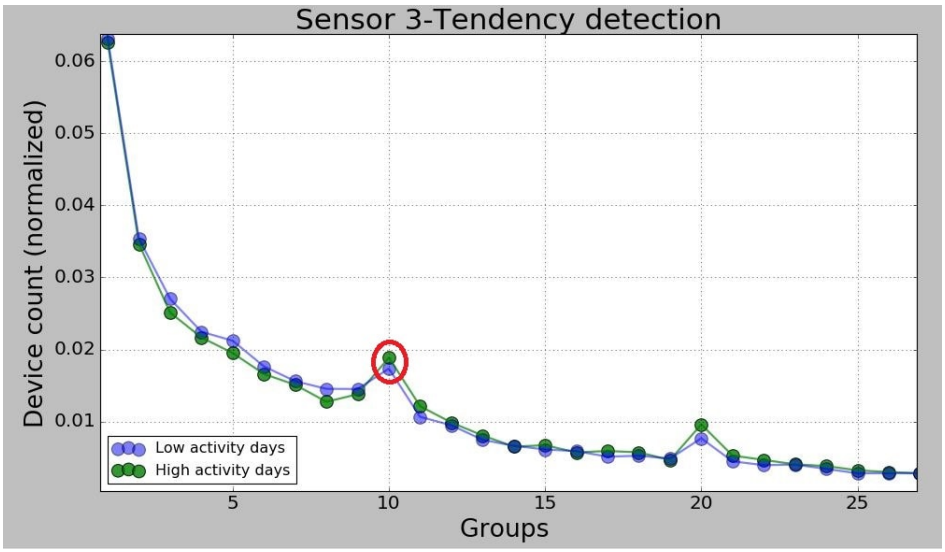


Figure 4.4: On sensor 2, there are more devices on low activity days until the Group 10, from which the tendency is reversed and then, there are more devices on high activity days. General view above, zoomed in view below



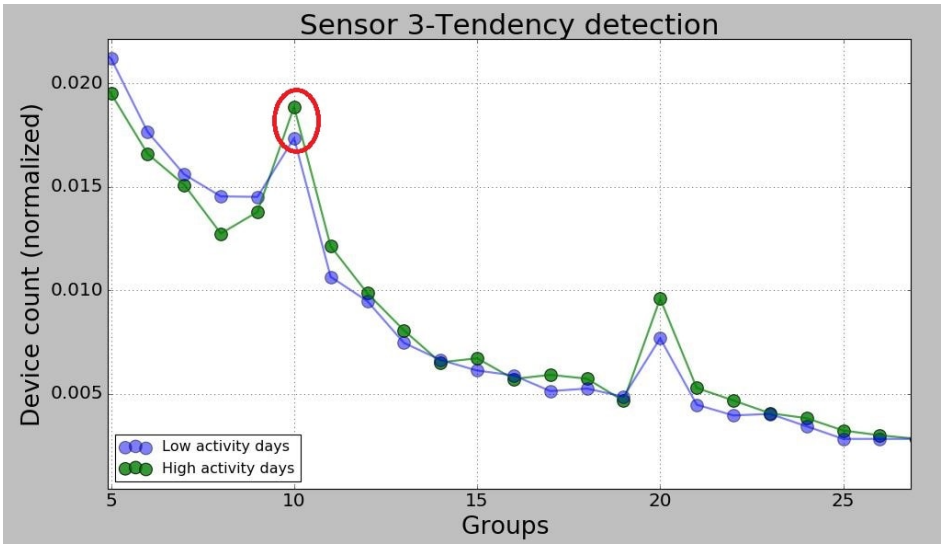


Figure 4.5: On sensor 3, there are more devices on low activity days until the Group 10, from which the tendency is reversed and then, there are more devices on high activity days. General view above, zoomed in view below

As seen on Table 4.2, the Sensor 1's tendency change starts in group 26, which means that it will be considered that the devices which stay for longer than 26 minutes are the long stayers. In the case of the sensors 2 and 3, the threshold is in the group 10.

	Tendency change point (minute)
Sensor 1	26
Sensor 2	10
Sensor 3	10

Table 4.2: Tendency change on people's stay times in each sensor

4.3.3 Rejection ratio

After having made the distinction between the short stayers and the long stayers, the calculation of the rejection points ratio begins. In this process, the general ratio of the number of new short stayers per each new long stayer is calculated, since it will inform of the moments in which there will be a higher amount of new short stayers than usual. That event would mean that the area is crowded or even full. This ratio enables the detection of the rejection points, which, consequently, confirms if a possible maximum capacity is the actual maximum or just the highest number recorded, and also calculates the threshold for distinguishing the crowded and not crowded states.

To calculate the mentioned ratio, it is searched for the moments in which the median of the short stayers in the high activity days is higher than the median of the short stayers in the low activity days. This is done because those moments are considered rejection intervals. Those moments are considered rejection intervals since they are of short duration and the expected behaviour on the high activity days would be to have less new people staying short periods of time than in low activity days, due to the necessity of the students to finish their tasks. Due to this fact, if there is a higher amount of new short stay devices at the high activity days, it means that something is keeping the people from staying inside that area, which would be the lack of space. The process to estimate the ratio is the following one:

Firstly, the ”busy threshold” is calculated to divide the moments which may have the possibility to be crowded and those which may not. To do so, the long stayers of the low activity days are retrieved and each day's highest number is used to make a mean, therefore obtaining the threshold.

Afterwards, the high activity days are traversed and the detected devices that entered at every minute are grouped, and their number is normalized by the total number of devices that were sensed on that day, in order to be able to make the comparison. After this, the intervals which have the same number of long stayers or more than the busy threshold are analysed. Those moments in which the median of the short stayers in high activity days is higher than the median of the long stayers are chosen as rejection intervals (Figures 4.6,4.7 and 4.8).

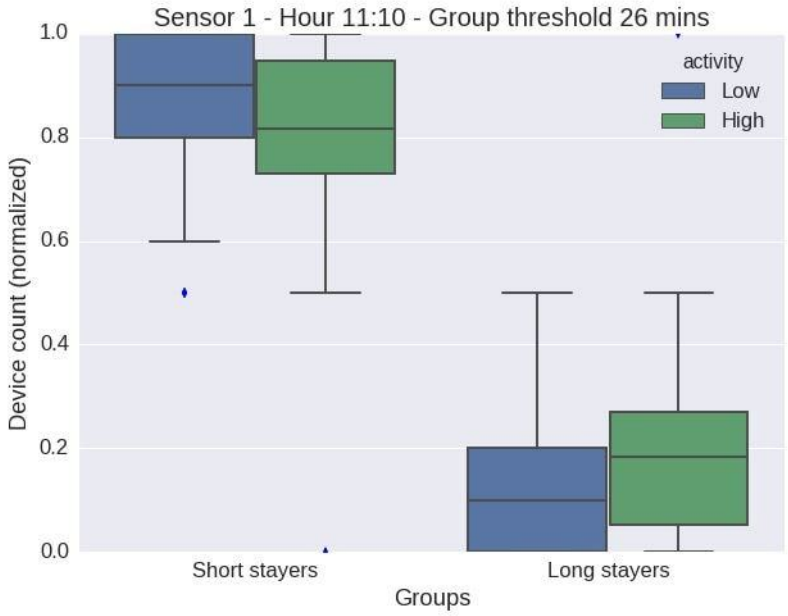


Figure 4.6: Rejection interval example at sensor 1

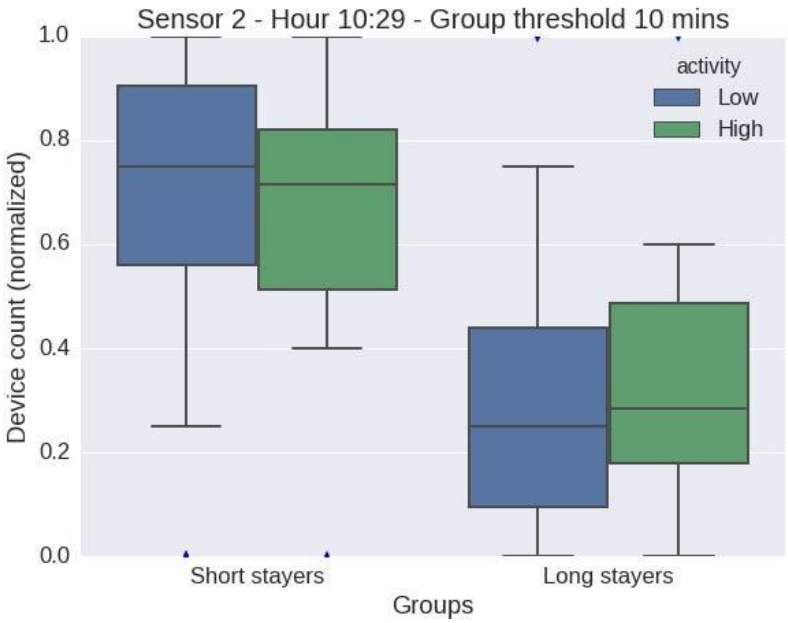


Figure 4.7: Rejection interval example at sensor 2

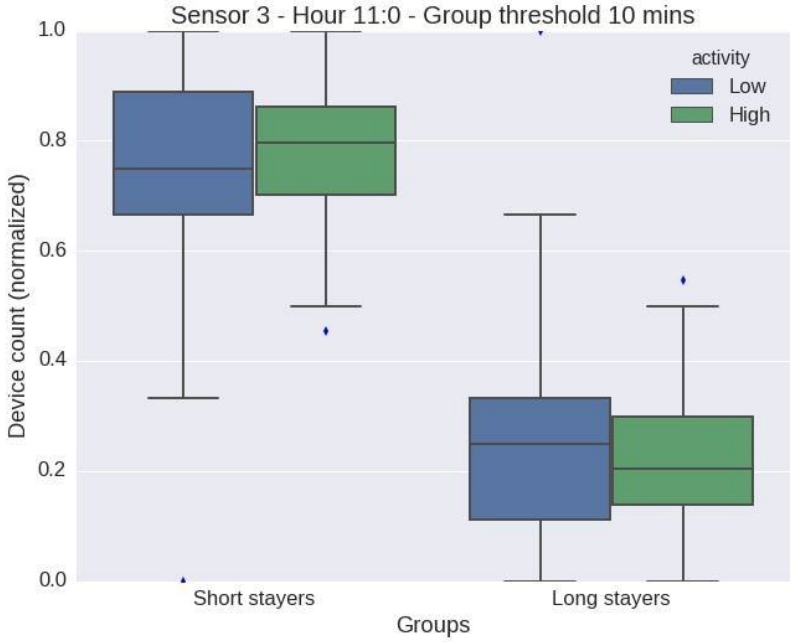


Figure 4.8: Rejection interval example at sensor 3

Also, in order to verify that the two types of medians are statistically different, a T-test is done in each sensor. This test is carried out since it is necessary to know if the two types of medians statistically different from each other. If they were not, different medians would be compared in the process but the results would be meaningless, which would imply that this approach is not appropriate to find the rejection inter-

vals. Due to this fact, a null hypothesis is formulated stating that the medians of the normed short stayers of the high activity days and the medians of the normed short stayers of the low activity days are not statistically different. Since the p-value in each of the T-Tests was inferior to 0.05 (Table 4.3), it means that the null hypothesis is discarded, meaning that the two types of medians are statistically different.

	p-value
Sensor 1	$6.27e^{-27}$
Sensor 2	$4.15e^{-9}$
Sensor 3	$8.37e^{-42}$

Table 4.3: p-value of the medians in each sensor

Finally, the detected rejection points are used to calculate the ratio of the number of short stay devices per long stay device should be detected to consider whether a rejection point is happening. To do so, on each rejection point, the number of short stayers on the high activity days is divided by the number of long stayers. Once this is done on each of the points, their ratios are averaged obtaining one general ratio for each sensor.

4.3.4 MACs to people conversion

As it was mentioned before, the sensors used detect devices capable of connecting to a Wi-Fi. Due to the popularity of the smartphones, tablets, laptops and other devices, these readings indicate the presence of people in an area. However, as a person can have no devices, one device, or more than one device, it is not possible to assume that for each sensed device there is one person. In consequence, it is necessary to find a way to translate the detected MAC addresses to the number of people in the area. In order to accomplish this task, the use of a Linear Regression algorithm has been chosen.

The linear regression algorithm fits a line to be able to predict, from the detected number of readings, the number of people. It needs some data to be able to trace the best fitting line (training phase) and for this means, the readings gathered in the ”Detected_addresses” group and the obtained ground truth are used. The ”Detected_addresses” group is used instead of the long stayers as, when the counting of people was carried out on each of the hours (10:00, 11:30, 14:00, 19:00 and 21:00), it is not known whether the people that were counted included people only from long stayers or from both groups. In addition, as the ground truth was collected by people, an error was made when the information was written down. Even though the

information was supposedly obtained at certain time (e.g.: 11:30) the most probable scenario is that the counting of people in one area started a little before and/or after that supposed time.

To correct these errors, a window of time will be used. It will go from 0 minutes to the maximum possible minute of one day (1439 minutes) in order to cover every minute of the day, independently from the time in which the recollection of data was made. During each window, as shown in the Algorithm 11 a lower limit and upper limit are calculated and the mean of the detected readings between those times is obtained. Then, the Pearsons correlation between the averaged readings and the people counted is calculated and stored with the variables used for its calculation.

Algorithm 11 Process for the compensation of the human error made when counting

```
Input: counted_people, countings's_time
Output: detected_readings_people

correlation_list = []
averaged_detected_readings_list = []

for (window=0; window<1440; w++)
    start_time = counting's_time  window
    stop_time = counting's_time + window
    detected_readings = get_detected_readings_in_window (start_time,
        stop_time)
    averaged_detected_readings = mean(detected_readings)
    correlation = get_correlation (averaged_detected_readings,
        counted_people)
    correlation_list.append (correlation)
    averaged_detected_readings_list.append(averaged_detected_readings)

picked_reading_number = get_first_peak_correlation(correlation_list,
    averaged_detected_readings_list)
return picked_reading_number
```

After processing all the possible windows, the window with the first peak is selected as the one to be used for that time. The first encountered peak is chosen instead of the highest correlation, as it is possible that the correlation of a non-reasonable window could be higher than the real one. For instance, this can be seen on Figures

4.9, 4.10 and 4.11, where the first peak is located in small time windows, but then, there are higher correlation values in very large windows. Nevertheless, the latter values should not be taken as indication of the duration of the ground truth gathering process as it is not reasonable to last 600 minutes over or 600 minutes under to supposed hour on which the data was obtained. However, it is more reasonable to assume that the gathering lasted 10 minutes more or 10 minutes less.

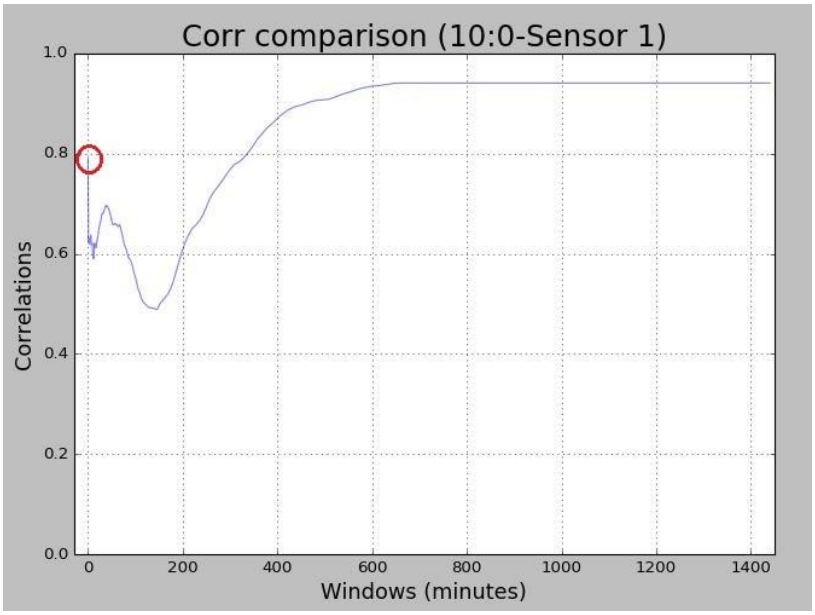


Figure 4.9: Comparison of the correlations in each window on sensor 1 at 10:00 (first peak marked in red)

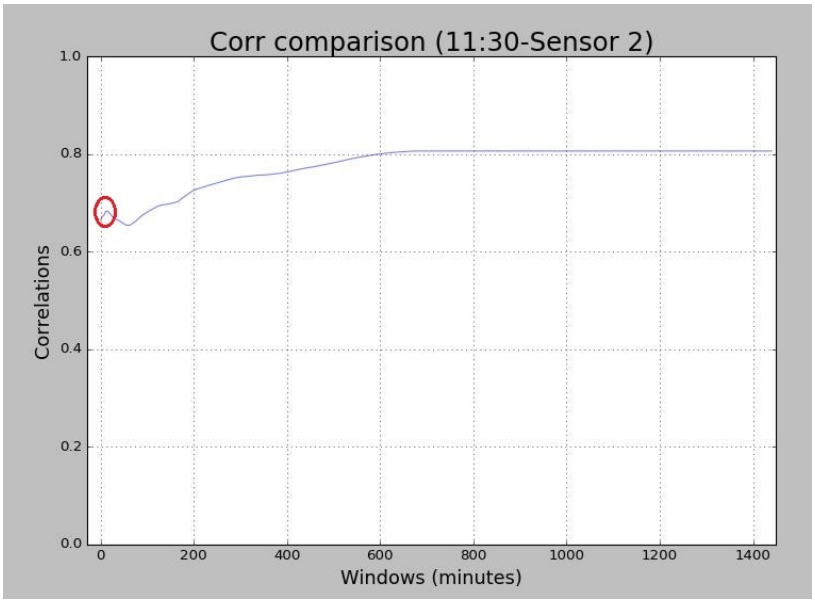


Figure 4.10: Comparison of the correlations in each window on sensor 2 at 11:30 (first peak marked in red)

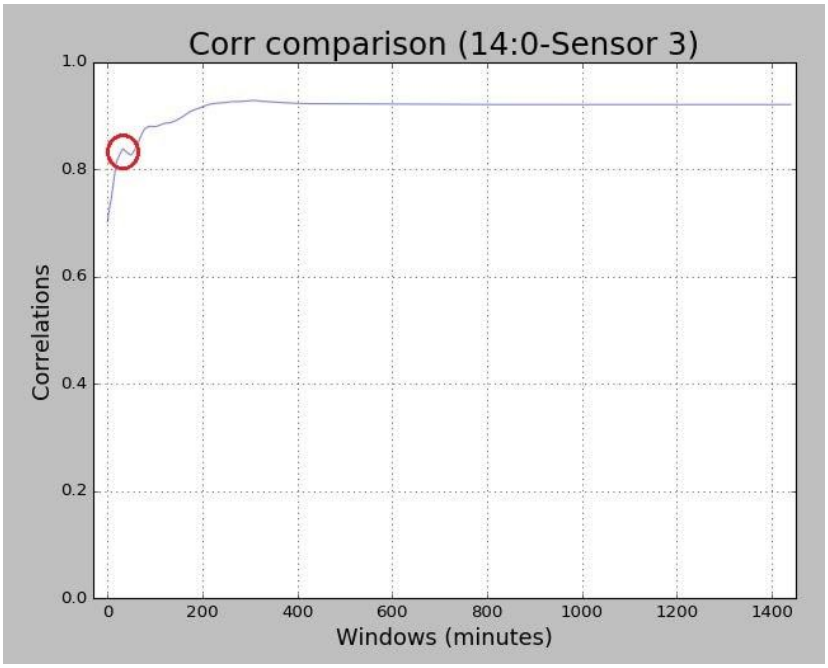


Figure 4.11: Comparison of the correlations in each window on sensor 3 at 14:00 (first peak marked in red)

Once the human error has been compensated for, the ground truth and the chosen averaged number of readings of the chosen windows are used to train a linear regression model. Also, it has to be taken into account that on each of the sensors, there are approximately 80 data samples of detected devices-counted people. As this number of samples is too low to appropriately train a model, a decision was made to feed with the data samples of all the sensors to a new linear regression model. This additional model will be linked to the implementation with an imaginary sensor labelled as Sensor 4.

To make sure that the final number of readings and the number of people in each sensor is correlated, the Pearsons correlation is measured on each of the sensors, obtaining the following values:

	Pearson's correlation
Sensor 1	0.86
Sensor 2	0.86
Sensor 3	0.85
Sensor 4	0.89

Table 4.4: Pearson's correlation between the sensed devices and the counted people in each area

However, even if there is a high correlation on each sensor, there may values in the

data which are very far from the reality (e.g.: moments in which there are 50 people and 0 readings were made), those points are called outliers. They can interfere in the training phase of the models, and to avoid it, those misleading points must be removed.

4.3.4.1 Outlier detection

If the average number of readings of the chosen windows are confronted with the counted people, the points representing the observed response (Figure 4.12) can be obtained. The first step to finding those points which are wrong or are unusual, is to create the best fitting line which will return the predicted values (Figure 4.12).

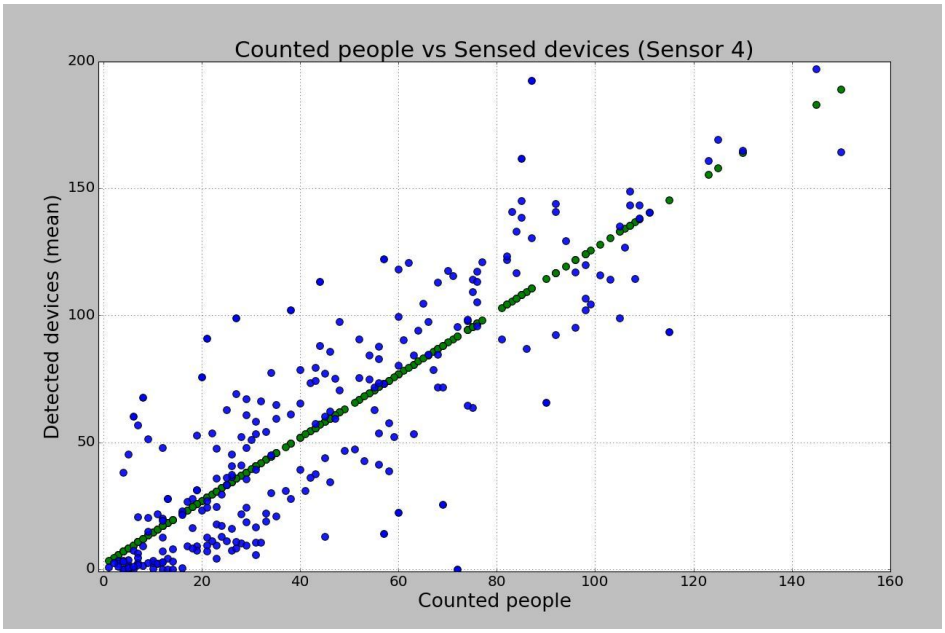


Figure 4.12: Observed responses (blue) and predicted values (green)

This line has a coefficient and an intercept that reduces the Residual Sum of Squares (RSS) to the minimum. The residuals express the distances between the observed response and the predicted value, such as:

$$e_i = y_i - \hat{y}_i \tag{4.1}$$

Where e_i is the i-th residual, y_i represents the i-th observed response and $\hat{y}_i$. represents the i-th predicted value.

The following step would be the calculation of the Studentized residuals of each point of the dataset. In order to obtain those value, the residuals, the leverage of each point, and the Mean Squared Error of that line are necessary.

The leverage of a point quantifies the influence that the observed response y_i has on its predicted value $\hat{y}_i$. If the leverage, h_{ii}, is small, then the observed value has a small role in the value of the predicted response $\hat{y}_i$, and vice versa. The h_{ii} values are measurements of the distances between the x value of the i-th data and the mean of the x values of the dataset, they have values between 0 and 1 (both inclusive) and the sum of all the h_{ii} values is equal to the number of coefficients that are used to obtain the line of best fit, including the intercept. The h_{ii} can be obtained using the formula:

$$h_{ii} = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{k=1}^{n} (x_k - \bar{x})^2} \tag{4.2}$$

Where x_i represents the x's i-th observed response, $\bar{x}$ is the mean of all the x values of the observed responses and the n is the number of observed responses.

The Mean Squared Error (MSE) measures how close a fitted line is to data points. The smaller the MSE, the closer the fitted line is to the data. It is calculated using the following formula:

$$MSE = \frac{1}{n} * \sum_{k=1}^{n} (\hat{y}_i - y_i)^2 \tag{4.3}$$

Where n is the number of data points and $\hat{y}_i$ is the i-th predicted value and y_i represents the i-th observed response.

Returning to the main subject, the Studentized residuals are calculated using the following formula:

$$r_i = \frac{e_i}{\sqrt{MSE(1 - h_{ii})}} \tag{4.4}$$

Once the Studentized residuals values are obtained, the points where the absolute values are greater than 2 are considered outliers and they will be discarded. This may seem quite arbitrary, but it is generally considered that attention should be paid to those values greater than |2|and even more to the ones greater than |3|(Figure 4.13). Finally, the remaining data is used for the training and testing of the Linear Regression model.

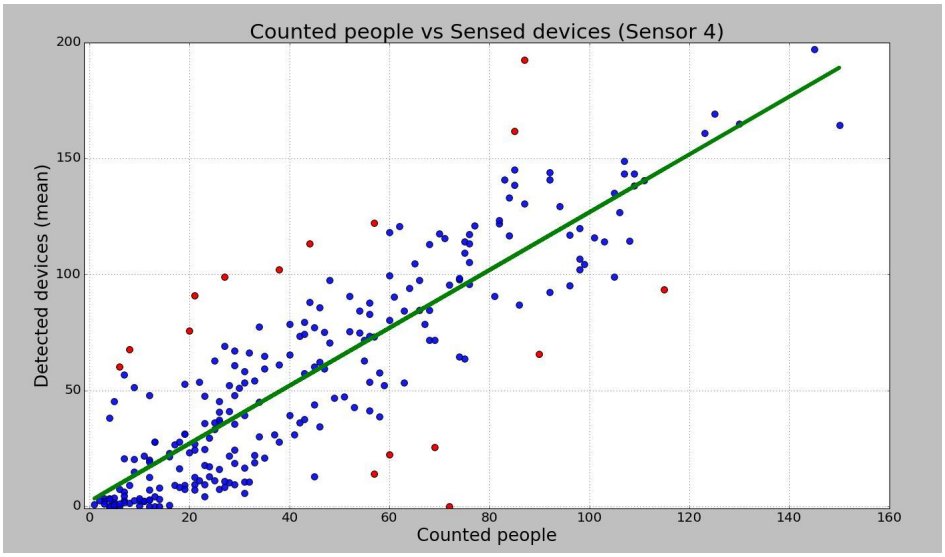


Figure 4.13: Example of outlier detection. Outliers (red), non-outliers (blue), best fitting regression line (green)

4.4 Results

In the following section, the results obtained in the validated areas are shown.

4.4.1 MAC to people conversion

The results of the linear regression models used for the MAC to people conversion have been evaluated using a 5-Fold Cross Validation algorithm to evaluate the average performance that each of the models have, depending on the training data. Instead of the usual 10 folds, 5 are used as with 10, the test groups would be very small due to the fact that in the sensors 1 to 3 there are only approximately 80 samples for training and testing.

Firstly, the R-Squared value of each sensor's model is calculated. As seen in the Figure 4.14, the model of the Sensor 1 is able to predict 20.27% of the test data, the model of the Sensor 2 is able to predict 44.93% and the model of the Sensor 3 can predict 30.54%. All of these models have a poor performance. However, the model of the imaginary Sensor 4, which was trained with the data of all the models, shows a performance of 69.98%. As the last model has the highest reliability, it is used for the MAC to people transformations.

In addition to the R-Squared value, the Root Mean Squared Error (RMSE) is also calculated in order to know the error margin in the predictions (Figure 4.15).

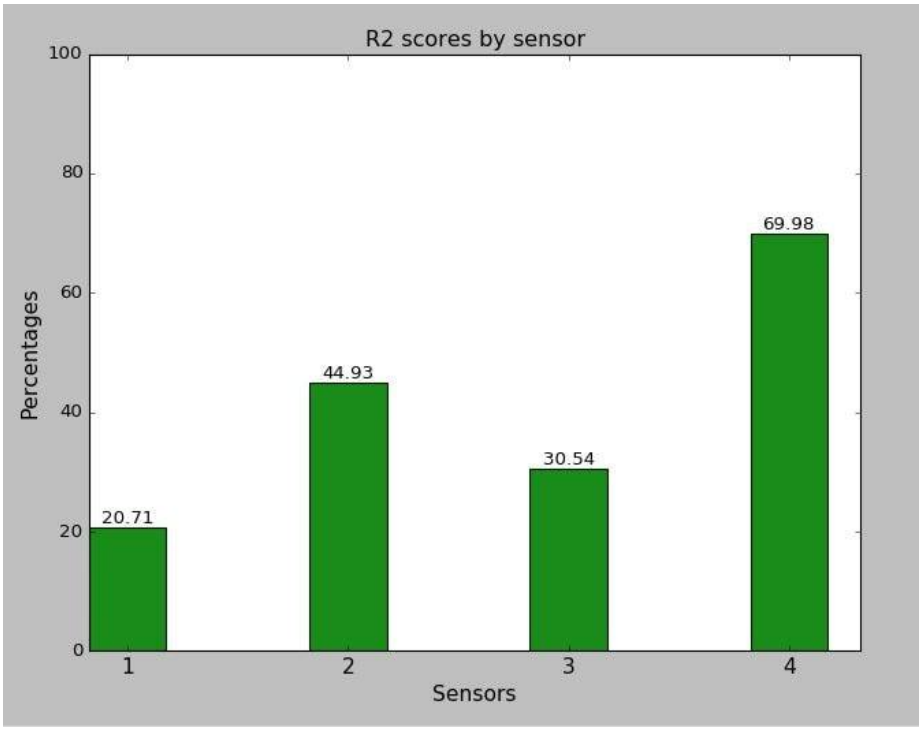


Figure 4.14: Linear regression models' R-Squared values

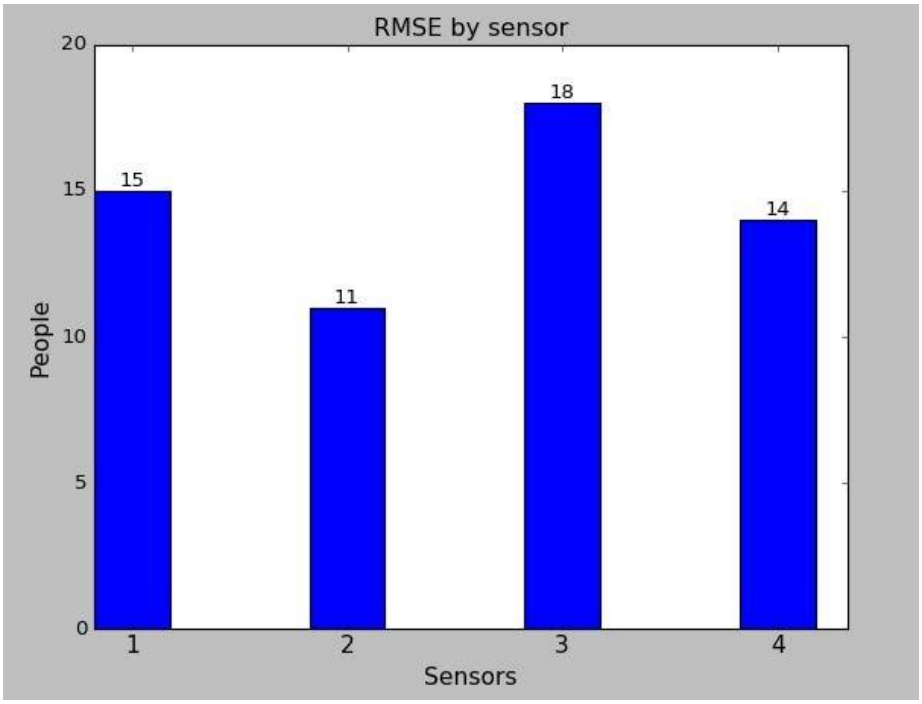


Figure 4.15: Linear Regression models' RMSE values

It is worth noting that even if the Sensor 4 has a good R-Squared mark, the gathering of more training data is required in order to create a more reliable regression model for each sensor.

4.4.2 Maximum capacity

To get the maximum capacity of an area, as it was indicated in the methodology, the highest number of long stayers in each area is obtained. These numbers are the possible maximum capacity of each area and, to confirm that they are the actual maximum capacities, on each of them, a rejection point with zero new long stayers should be found. This event should happen at the detected possible maximums as, if the maximum was truly achieved, new people should not be able to stay for a long time, for which a rejection point with zero new long stayers should be found.

In consequence, the long stayers of the high activity days and the rejection points are visualized in order to find out if in a possible maximum there is a rejection point with zero new long stayers. From the data, the following possible maximums are obtained (Figures 4.16,4.17, 4.18): 158 for the area of sensor 1, 172 for the area of sensor 2, and 203 for the area of sensor 3. However, they could not be confirmed as the actual maximum capacity since there were no rejection points on the possible maximums at sensors 1 and 3 (Figure 4.19 and 4.21) , and in sensor 2, even though there was a rejection point, at that moment, 7 new long stayers (Figure 4.20) were able to enter the area.

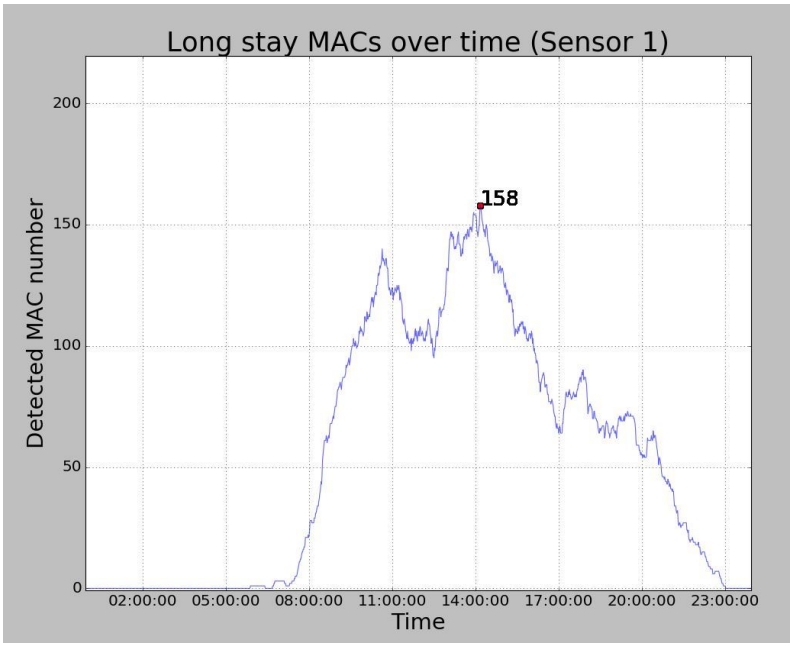


Figure 4.16: Possible maximum capacity in Sensor 1

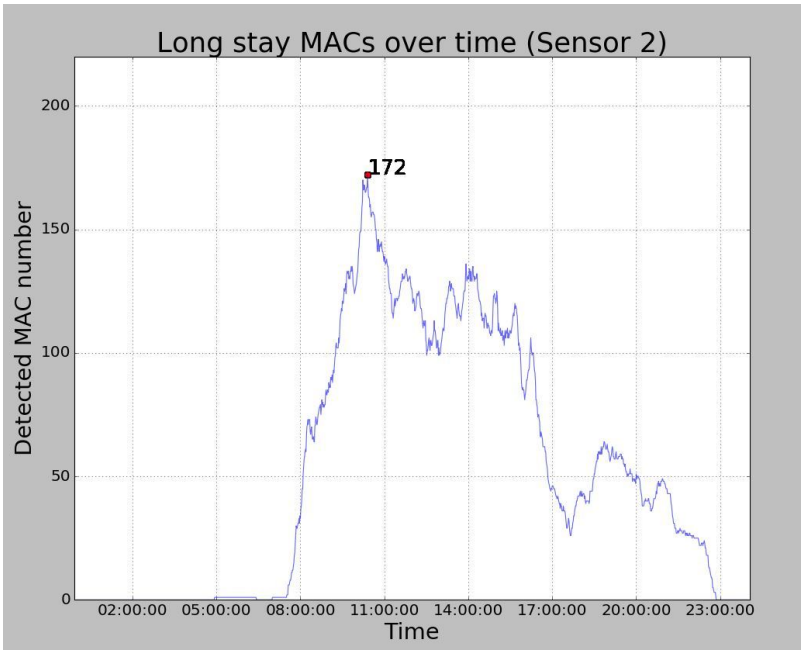


Figure 4.17: Possible maximum capacity in Sensor 2

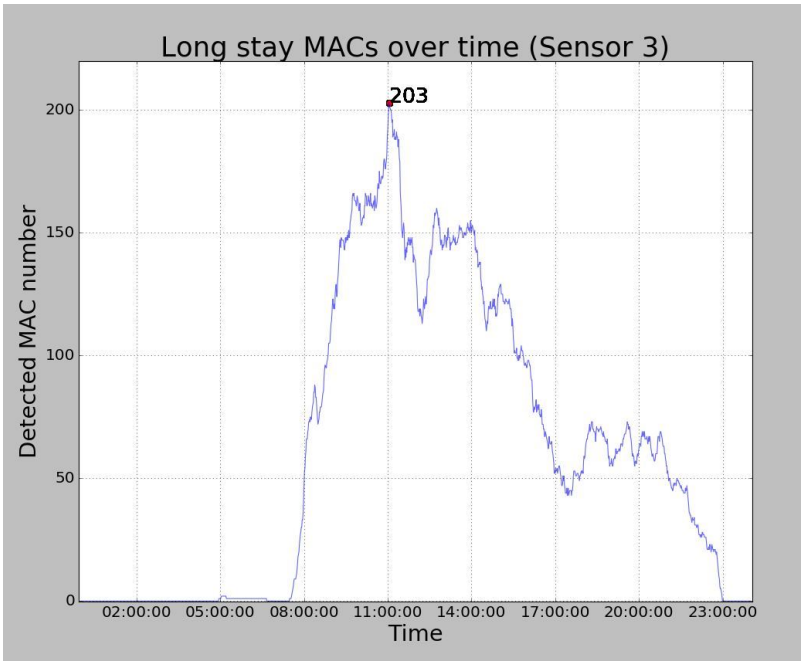


Figure 4.18: Possible maximum capacity in Sensor 3

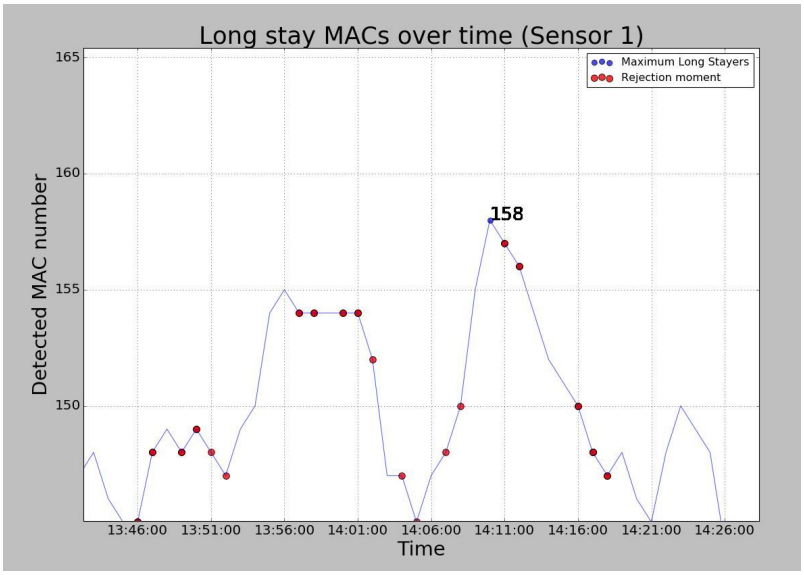


Figure 4.19: Rejection points close to the possible maximum in Sensor 1

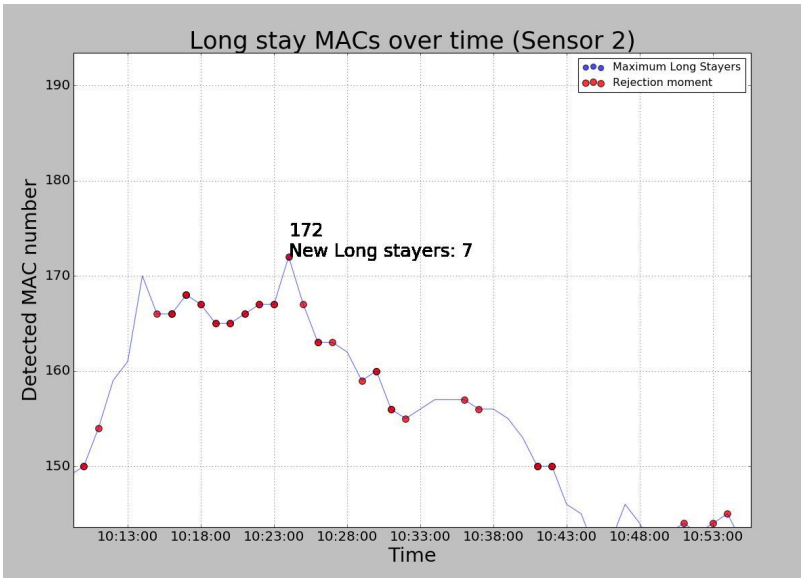


Figure 4.20: Rejection points at the possible maximum in Sensor 2

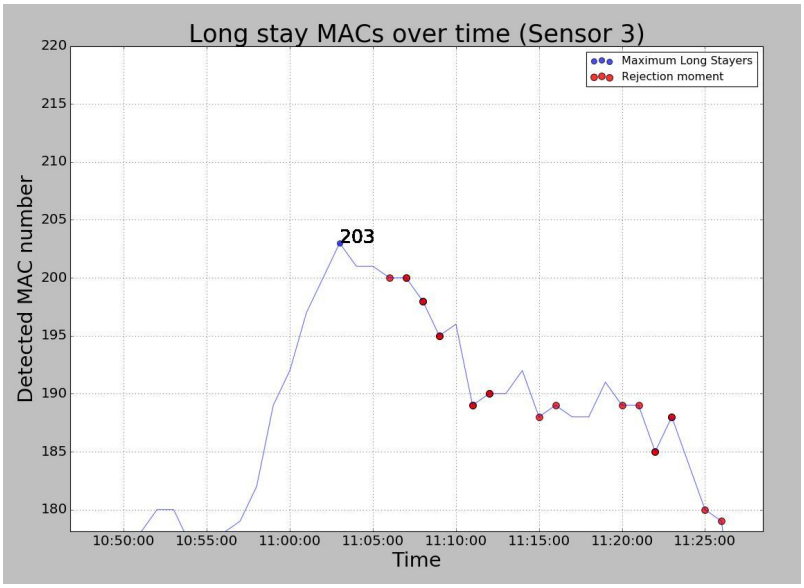


Figure 4.21: Rejection points close to the possible maximum in Sensor 3

After transforming the possible maximum capacity in MACs to numbers of people using the linear regression model of the Sensor 4, it was obtained that the maximum capacity for the area covered by the sensor 1 is 109.81, the maximum for the sensor 2 is 118.87 and the maximum for the sensor 3 is 138.92. However, as those numbers represent people, they are rounded up to the nearest integer value (Table 4.5).

	Maximum in MACs	**Maximum in people**	**Error margin**
Sensor 1	158	$109.81 \approx 110$	14
Sensor 2	172	$118.87 \approx 119$	14
Sensor 3	203	$138.92 \approx 139$	14

Table 4.5: Predicted maximum capacities

The conversion of the maximums from MAC to people numbers enables the comparison of the calculated capacity to the available number of seats at each zone (it is considered that 1 person will occupy 1 seat). However, the comparison between the predicted values and the available seats is challenging. The reason for this challenge is that in the analysed areas there are individual seats and several rooms destined for doing group assignments (these are called ”Project rooms”).

In each area, there is a different number of project rooms and not all of them have the same number of seats. In the following table, the number of project rooms in each area are shown, with the minimum and maximum seats counted inside each area’s project rooms:

	Project rooms	Minimum seats	Maximum seats
Area 1	5	6	6
Area 2	12	6	10
Area 3	15	6	10

Table 4.6: Project rooms and seats in each area

The problem appears when a room is being used, since the group that is in it may not use the whole room, but the room will be occupied as there can only be one group per room. In consequence, in the case of having every other room and the individual seats occupied, the zone will be at its full capacity even if all the seats are not occupied. As it is also highly unlikely to have all the project rooms occupied at their full capacity along with the individual seats, a decision was made to estimate the maximum capacity in two ways: the first estimation will be made considering all the counted seats (this will be identified as the ”high estimation”) while the second one will be made considering that the project rooms have a maximum of 6 seats. The reason for using 6 seats as a maximum is because it is not usual to be in groups larger than 6 people.

As seen in Figure 4.22, if the predictions are compared with either of the estimations, there is a difference of, at least, 12 seats. This could be caused by the error margin during the conversion from MACs to people numbers (marked in red on the graph) and also by the fact of not having all the project rooms occupied at their full capacity.

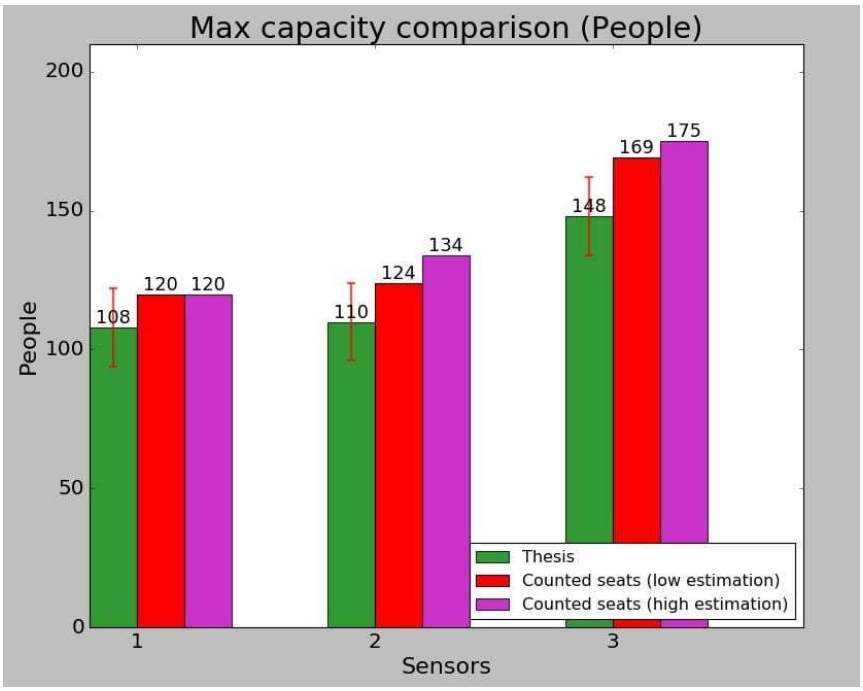


Figure 4.22: Maximum capacity comparison

4.4.2.1 Baseline

As it was explained in the Background (Chapter 2), only one research ([1]) was found in which the maximum capacity of an area was calculated. In consequence, it is used as baseline. In this study, the maximum capacity of an area is obtained by taking the highest number of detections encountered in the analysed period. However, this method does not include a process to discard the detections that may not belong to people using the area, nor a method to confirm that the highest number is the actual maximum and not only the highest number of devices recorded until that moment. Nonetheless, in order to observe the results that this approach would have in this thesis' scenario, it has been tested.

Following the procedure of the baseline, from the whole number of detections, the highest number is taken as a possible maximum capacity. This is done instead of distinguishing between short and long stayers. As seen in the Figure 4.23, the results obtained by this method seem to be similar to the ones achieved by the procedure implemented in this thesis, but instead of obtaining lower values than those considered to be the real ones, which is what is obtained in this thesis, higher values are obtained as possible maximum capacities. This is logical, due to the lack of a method to filter out the readings of devices that are detected by the sensor but not using the area.

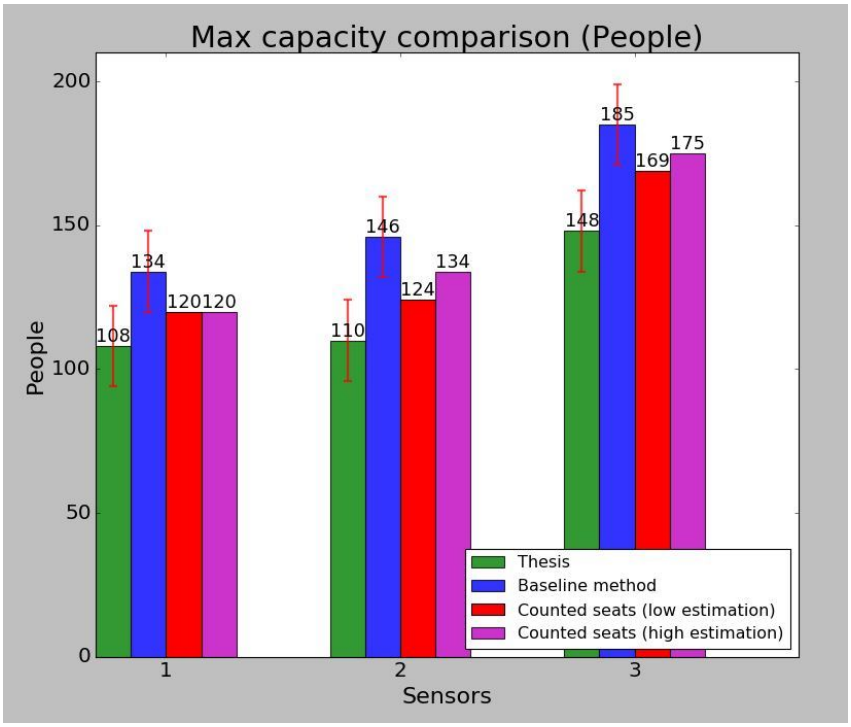


Figure 4.23: Comparison of the results obtained by the thesis' method with the one of the baseline

4.4.3 Crowdedness threshold

For the distinction of the not crowded and crowded moments, as explained in the Methodology Background (Chapter 3), the number of long stayers in each of the detected rejection points in one day are averaged obtaining a threshold for that day. However, as a general threshold is desired, the thresholds of all the days are averaged to obtain one final threshold which, if its surpassed, will lead to the area to be considered crowded. As seen in Figures 4.24, 4.25 and 4.26, the general crowdedness threshold for Sensors 1,2 and 3 are 101.81,122.08 and 146.57, respectively. If those device numbers are transformed into people, the thresholds become 73.58, 87.17 and 102.69. However as those numbers represent people, they are rounded up being considering that the thresholds are 74, 88 and 103 (Table 4.7).

	Crowdedness threshold in MACs	**Crowdedness threshold in people**	**Error margin**
Sensor 1	101.81	$73.58 \approx 74$	14
Sensor 2	122.08	$87.17 \approx 88$	14
Sensor 3	146.57	$102.69 \approx 103$	14

Table 4.7: Predicted general crowdedness thresholds

If a comparison with the estimated maximum capacities is made (Figure 4.27), then it is found that the obtained limits seem to be reasonable. However, due to time issues, it was not possible to confirm that when the number of people was close to or above those limits, the expected behaviour of people of not staying in the area for long times due to the crowdedness was happening, which would confirm the validity of the calculated thresholds.

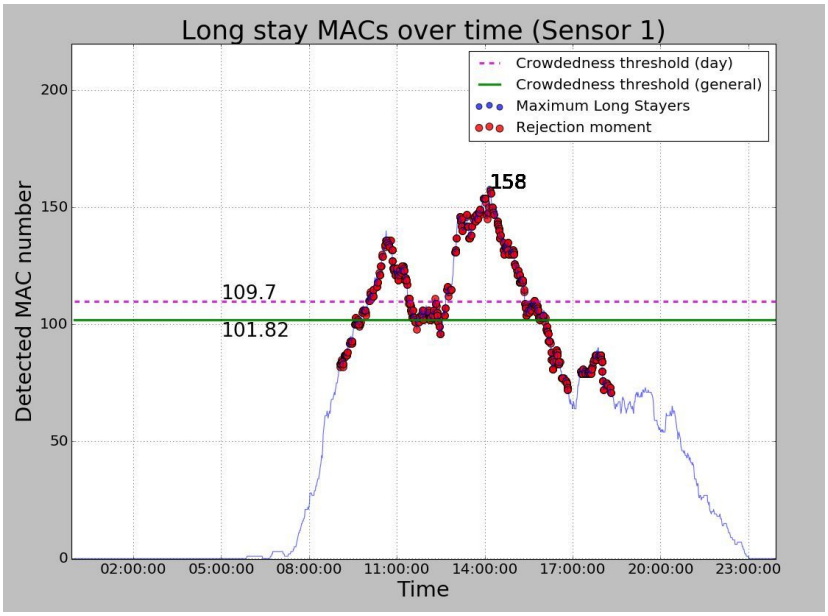


Figure 4.24: Crowdedness thresholds in Sensor 1

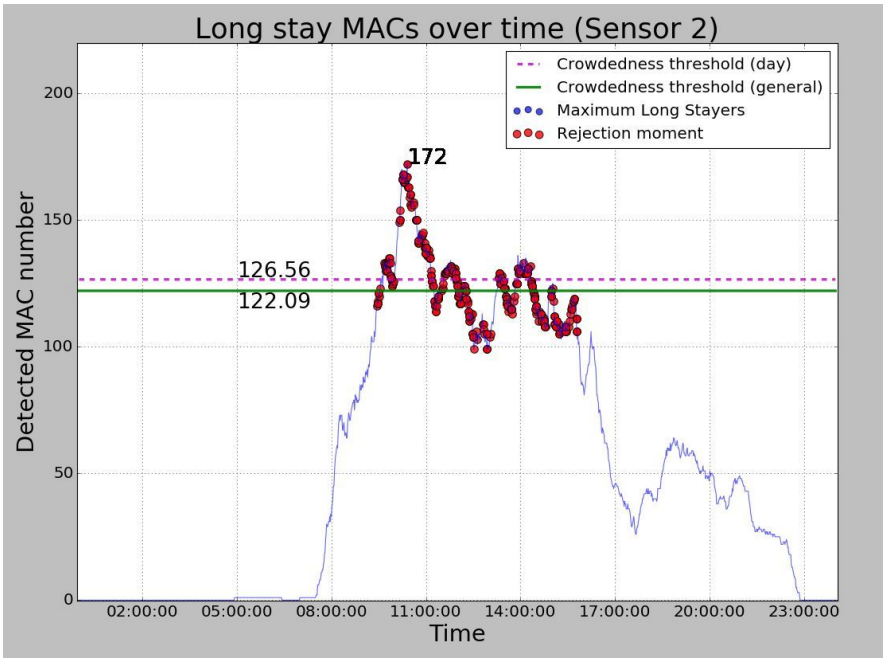


Figure 4.25: Crowdedness thresholds in Sensor 3

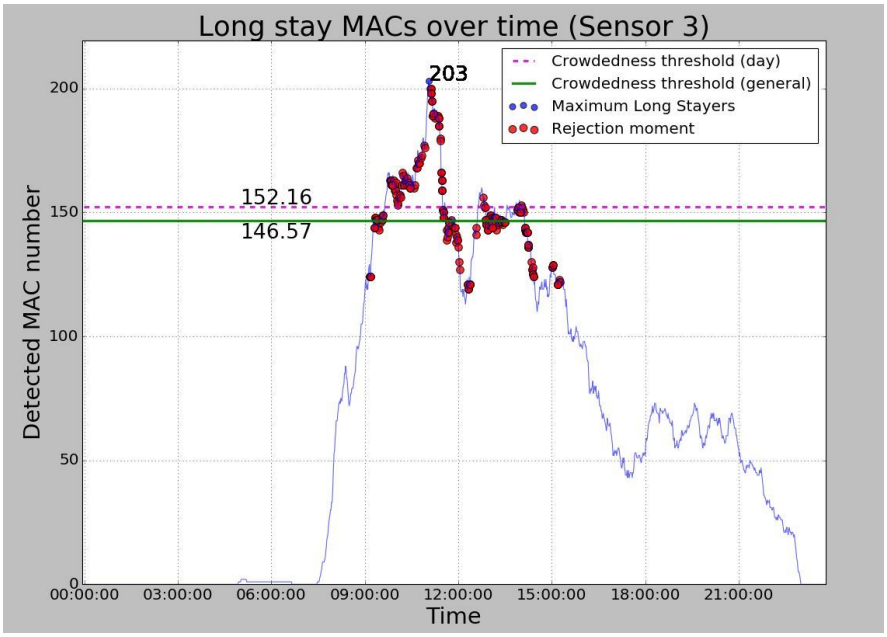


Figure 4.26: Crowdedness thresholds in Sensor 3

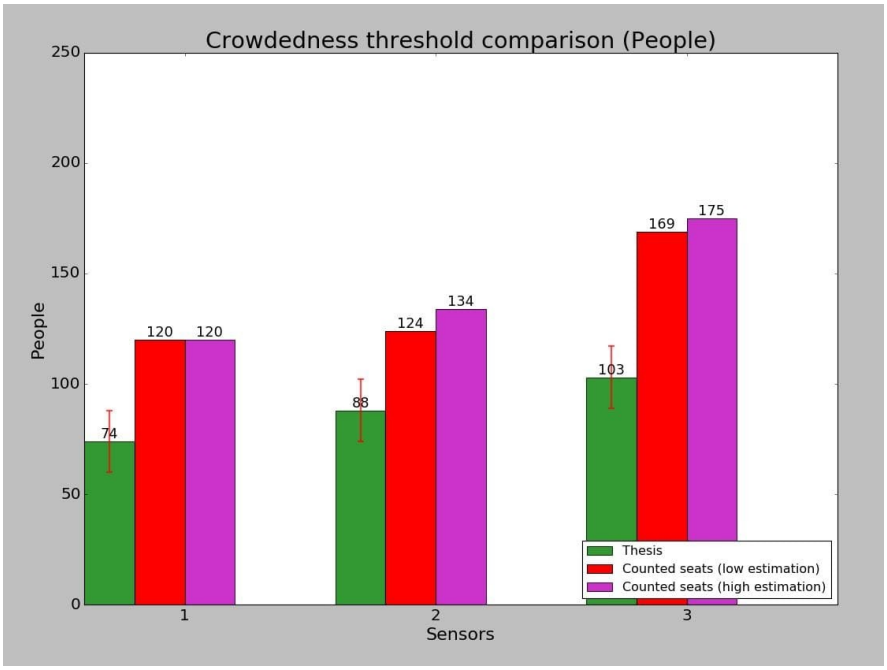


Figure 4.27: Crowdedness threshold comparison with the considered maximum capacities

Chapter 5

Conclusions and Future work

5.1 Conclusions

This thesis describes the design of a method for the detection of crowded places in which the people have predefined seats. The design seeks to know the number of people in the analysed place, its maximum capacity and the threshold that distinguishes the states of ”not crowded” and ”crowded”. The evaluation of the methods was performed in a real context (a library) in three different areas, each one covered by a sensor.

The calculation of people in the different zones, using the number of devices in each of them, was validated, and the results showed that the used approach was adequate. This was done by obtaining an R-Squared value of 69% in the linear regression which is used to predict from the detected devices, the number of people. Nevertheless, since it was necessary to combine the gathered ground-truth samples from the three areas to create a reliable model, and as its reliability could be improved, it is concluded that more ground-truth data should be collected.

Regarding the calculation of the maximum capacity of an area, an algorithm was described to estimate the maximum capacity of a place taking into account the errors introduced by the devices passing close to the analysed zone but not using it, which was not found in the existing literature, which is very limited. The results obtained from the validation of this algorithm were found to be inconclusive as, due to time issues, it was not possible to visually confirm that the areas were full at any moment during the considered high activity days. This entails that even if a possible maximum was calculated by the algorithm, it was not possible to confirm that it was the actual maximum.

Finally, a novel algorithm for the automatic calculation of the threshold for the dis-

tinction of the ”crowded” and ”not crowded” states was designed for areas with predefined seats for people, which until now had not be done, since the threshold for that state distinction was arbitrarily established. In addition, the results of this project were considered reasonable but their practical value could not be confirmed since, as before, there were time issues and it could not be visually confirmed.

5.2 Future work

As mentioned, this approach considers that the analysed place has high and low activity days, the former being the days in which it is more likely that the zone reaches its maximum capacity. In this project, those days are chosen based on the exam calendar of the university in which the library used for the validation is placed, however, the automatic classification of those days could be very useful for more exact calculations of the maximum capacity of an area in which the people tend to stay for long periods of time. For instance, this could help to remove days in which anomalies are detected and an odd number of readings are captured due to an unusual use of the analysed zone.

In addition, the behaviour of the people and their stay times when they enter into a crowded area has also been mentioned. A deeper research of that behaviour could gather the necessary information to be able to develop more efficient algorithms for the automatic calculation of the threshold for the ”not crowded” and ”crowded” states.

Bibliography

[1] I. Ceapa, C. Smith, and L. Capra, "Avoiding the crowds: understanding tube station congestion patterns from trip data," *... of the ACM SIGKDD International Workshop ...*, pp. 134–141, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2346518

[2] H. Chourabi, T. Nam, S. Walker, J. R. Gil-Garcia, S. Mellouli, K. Nahon, T. A. Pardo, and H. J. Scholl, "Understanding smart cities: An integrative framework," *Proceedings of the Annual Hawaii International Conference on System Sciences*, pp. 2289–2297, 2011.

[3] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, "Smart Cities and the Future Internet: Towards Cooperation Frameworks for Open Innovation The Future Internet SE - Lecture Notes in Computer Science," vol. 6656, pp. 431–446, 2011.

[4] J. Zhou, H. Pei, and H. Wu, "Early Warning of Human Crowds Based on Query Data from Baidu Map: Analysis Based on Shanghai Stampede," 2016. [Online]. Available: http://arxiv.org/abs/1603.06780

[5] D. Helbing and et al., "Crowd disasters as systemic failures: analysis of the Love Parade disaster," *EPJ Data Science*, vol. 1, no. 1, p. 7, 2012. [Online]. Available: http://www.epjdatascience.com/content/1/1/7

[6] K. Rainer and M. Ruhnke, "A Navigation System for Robots Operating in Crowded Urban Environments," pp. 3225–3232, 2013.

[7] L. Kratz and K. Nishino, "Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models," *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pp. 1446–1453, 2009.

[8] Y. Cong, J. Yuan, and J. Liu, "Abnormal event detection in crowded scenes using sparse representation," *Pattern Recognition*, vol. 46, no. 7, pp. 1851–1864, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2012.11.021

[9] G. Andrienko, N. Andrienko, C. Hurter, S. Rinzivillo, and S. Wrobel, “From Movement Tracks through Events to Places : Extracting and Characterizing Significant Places from Mobility Data,” pp. 161–170, 2011.

[10] N. Aschenbruck, A. Munjal, and T. Camp, “Trace-based mobility modeling for multi-hop wireless networks,” *Computer Communications*, vol. 34, no. 6, pp. 704–714, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.11.002

[11] L. Li, G. Shen, C. Zhao, T. Moscibroda, J.-h. Lin, and F. Zhao, “Experiencing and Handling the Diversity in Data Density and Environmental Locality in an Indoor Positioning Service Categories and Subject Descriptors.”

[12] J. Krumm and J. Platt, “Minimizing Calibration Efforts for an Indoor 802.11 Device Location Measurement System,” *Measurement*, pp. 1–9, 2003. [Online]. Available: http://eprints.pascal-network.org/archive/00000067/

[13] P. Bahl and V. N. Padmanabhan, “RADAR: An In-building RF-based User Location and Tracking System,” *Proc. IEEE INFOCOM 2000. The 19th annual conference on Computer Communications*, vol. 2, pp. 775–784, 2000.

[14] M. A. Youssef and A. Agrawala, “The Horus WLAN location determination system,” *Proceedings of the 3rd international conference on Mobile systems, applications, and services - MobiSys ’05*, pp. 205–218, 2005. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1067170.1067193%5Cnhttp://dl.acm.org/citation.cfm?id=1067170.1067193

[15] L. F. M. de Moraes and B. A. a. Nunes, “Calibration-free WLAN location system based on dynamic mapping of signal strength,” *Proceedings of the international workshop on Mobility management and wireless access - MobiWac ’06*, pp. 92–99, 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1164783.1164799

[16] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, “Indoor localization without the pain,” *Proceedings of the sixteenth annual international conference on Mobile computing and networking - MobiCom ’10*, p. 173, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1859995.1860016

[17] H. Lim, L. C. Kung, J. C. Hou, and H. Luo, “Zero-configuration, robust indoor localization: Theory and experimentation,” *Proceedings - IEEE INFOCOM*, pp. 1–13, 2006.

[18] R. Tesoriero, R. Tebar, J. A. Gallud, M. D. Lozano, and V. M. R. Penichet, “Expert Systems with Applications Improving location awareness in indoor

spaces using RFID technology," *Expert Systems With Applications*, vol. 37, no. 1, pp. 894–898, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2009.05.062

[19] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[20] C. Mascolo, "Designing Mobility Models based on Social Network Theory," vol. 1, no. 2, pp. 1–12.

[21] X. Z. X. Zhao, E. Delleandrea, and L. C. L. Chen, "A People Counting System Based on Face Detection and Tracking in a Video," *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, no. July, pp. 67–72, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5279466

[22] T. van Oosterhout, S. Bakkes, B. Krse, T. V. Oosterhout, and B. Kr, "Head Detection in Stereo Data for People Counting and Segmentation," *Proceedings of the International Conference on Computer Vision Theory and Applications*, no. 2003, pp. 620–625, 2011.

[23] M. Li, Z. Zhang, K. Huang, and T. Tan, "Estimating the Number of People in Crowded Scenes by MID Based Foreground Segmentation and Head-shoulder Detection," 2008.

[24] H. Xu, L. Pei, and M. Lei, "A people counting system based on head-shoulder detection and tracking in surveillance video," *2010 International Conference on Computer Design and Applications, ICCDA 2010*, vol. 1, no. Iccda, pp. 394–398, 2010.

[25] C. Zeng and H. Ma, "Robust head-shoulder detection by PCA-based multilevel HOG-LBP detector for people counting," *Proceedings - International Conference on Pattern Recognition*, pp. 2069–2072, 2010.

[26] Y. P. Raykov, E. Ozer, G. Dasika, A. Boukouvalas, and M. A. Little, "Predicting room occupancy with a single passive infrared (PIR) sensor through behavior extraction," *UbiComp*, pp. 1016–1027, 2016.

[27] M. Zhou, M. Ma, Y. Zhang, K. Sui, D. Pei, and T. Moscibroda, "EDUM: Classroom education measurements via large-scale WiFi networks," *UbiComp 2016 - Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2016.

[28] C. Xu, S. Li, G. Liu, and Y. Zhang, “Crowd ++: Unsupervised Speaker Count with Smartphones,” *Ubicomp*, pp. 43–52, 2013.

[29] M. B. Kjaergaard, M. Wirz, D. Roggen, and G. Troster, “Mobile sensing of pedestrian flocks in indoor environments using WiFi signals,” *2012 IEEE International Conference on Pervasive Computing and Communications, PerCom 2012*, no. March, pp. 95–102, 2012.

[30] M. Kjaergaard, M. Wirz, D. Roggen, and G. Troster, “Detecting pedestrian flocks by fusion of multi-modal sensors in mobile phones,” *ACM Conference on Ubiquitous Computing*, pp. 240–249, 2012.

[31] H. Rahmalan, M. S. Nixon, and J. N. Carter, “On Crowd Density Estimation for Surveillance,” *The Institution of Engineering and Technology Conference on Crime and Security*, pp. 540–545, 2006. [Online]. Available: http://eprints.soton.ac.uk/262852/

[32] G. Kim, T. An, and M. Kim, “Estimation of Crowd Density in Public Areas Based on Neural Neural Network,” *KSII Transactions on Internet and Information Systems*, vol. 6, no. 9, pp. 2170–2190, 2012.

[33] B. Anzengruber, D. Pianini, J. Nieminen, and A. Ferscha, “¡Predicting Social Density in Mass Events.pdf¿,” no. i, pp. 206–215, 2013.

[34] J. Weppner and P. Lukowicz, “Bluetooth based collaborative crowd density estimation with mobile phones,” *2013 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, no. March, pp. 193–200, 2013. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6526732%5Cnpapers3://publication/doi/10.1109/PerCom.2013.6526732

[35] Y. Maekawa, A. Uchiyama, H. Yamaguchi, and T. Higashino, “Car-level congestion and position estimation for railway trips using mobile phones,” *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp ’14 Adjunct*, pp. 939–950, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2632048.2636062

[36] K. Farkas, G. Fehér, A. Benczúr, and C. Sidló, “Crowdsending Based Public Transport Information Service in Smart Cities,” *IEEE Communications Magazine*, vol. 53, no. August, pp. 158–165, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/cm/cm53.html#FarkasFBS15

Appendix A

Group classification example

This example shows how the classification algorithm works using some invented MAC addresses and timestamps. In order to make the example easier, there are no randomized MAC's readings, the readings will be ordered by timestamp and treated as addresses directly, and they will also have, next to them, their Timestamp in brackets and the "Time Limit" will be 2 minutes.

A.1 Processed minute 1 (12:58)

As said, first the readings are recovered from the group with all the readings. Then, it will be checked if any of those addresses were present in the previous minutes:

Data set	
Time	**Addresses**
12:58	A (12:58) B (12:58) A (12:58)
12:59	
13:00	B (13:00)
13:01	

Addresses in previous minutes

As they were not present, their presence in the current minute will also be checked. In this case, the first two addresses were not present during this minute, so they are added to the "Addresses_in_this_minute" group. However, by the time that the third address is processed, the first "A" address is considered to be in this minute, so this new reading is discarded to avoid having two "A"s in the same minute, which would mean that the same device is counted twice. Then, the considered addresses to be in this minute are:

Addresses in this minute	
A (12:58)	B (12:58)

In addition, as the 2 different addresses were not found in the ”Addresses_in_previous_minutes” or in this minute, they are added to the ”New_addresses” group.

New addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	
13:00	
13:01	

Step 1 from the explanation is completed and we proceed to Step 2. However, as in this example, at this moment there are not any addresses in previous minutes, the checking of the ”Time outs” is not carried out. Afterwards, the addresses at ”Addresses_in_this_minute” are inserted into the ”Detected_addresses” group, in the position that corresponds to the time that was just analysed (12:58).

Detected addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	
13:00	
13:01	

Finally, the addresses in ”Addresses_in_this_minute” are copied to ”Addresses_in_previous_minutes” in order to empty the first group and be able to reuse it in the next processed minute.

Addresses in this minute

Addresses in previous minutes	
A (12:58)	B (12:58)

A.2 Processed minute 2 (12:59)

The second minute is processed. The addresses from this new minute are recovered and a check is carried out to find out if any of them are in ”Addresses_in_previous_minutes”.

Data set	
Time	**Addresses**
12:58	A (12:58) B (12:58) A (12:58)
12:59	
13:00	B (13:00)
13:01	

As there are no new addresses, the ”Addresses_in_previous_minutes” and ”Addresses_in_this_minute” contain the following data.

Addresses in this minute

Addresses in previous minutes
A (12:58) B (12:58)

Now, it is calculated if any of the previous addresses has timed out. To do so, the difference between the processed time and their timestamp is calculated:

$$12:59 - 12:58 = 1$$

$$1 < 2 \rightarrow \text{No Time out}$$

In this case, neither ”A” nor ”B” timed out, as their timestamp difference is inferior to the Time Limit. At this moment, the addresses that did not time out are copied into ”Addresses_in_this_minute” and the result is copied into ”Detected_addresses”.

Addresses in this minute
A (12:58) B (12:58)

Detected addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	A (12:58) B (12:58)
13:00	
13:01	

Finally, the addresses in ”Addresses_in_this_minute” are copied to ”Addresses_in_previous_minutes” and the first group is emptied.

Addresses in this minute

Addresses in previous minutes
A (12:58) B (12:58)

A.3 Processed minute 3 (13:00)

The readings of this new minute are recovered and checked if they were present in previous minutes. As in this case ”B” was, it is not included in the ”New_addresses”

group.

Data set	
Time	**Addresses**
12:58	A (12:58) B (12:58) A (12:58)
12:59	
13:00	B (13:00)
13:01	

Addresses in previous minutes
A (12:58) B (12:58)

Afterwards, as the ”B” address was present during previous minutes and now there is a more recent reading of it, it is deleted from the ”Addresses_in_previous_minutes” as its timestamp will be refreshed with this new reading.

Addresses in previous minutes
A (12:58) ~~B (12:58)~~

Now that the Step 1 is finished, Step 2 will begin. First it is checked whether A times out.

$$13:00 - 12:58 = 2$$

$$2 < 2 \rightarrow \text{Time out}$$

As indeed it does time out, it is added to the ”Gone_addresses” group at the moment indicated by the timestamp of the reading.

Gone addresses	
Time	**Addresses**
12:58	A (12:58)
12:59	
13:00	
13:01	

Now ”B”, which is the only address considered to be present in this minute, is copied into ”Detected_addresses”.

Addresses in this minute
B (13:00)

Detected addresses		
Time	**Addresses**	
12:58	A (12:58)	B (12:58)
12:59	A (12:58)	B (12:58)
13:00		B (13:00)
13:01		

Finally, the addresses in ”Addresses_in_this_minute” are copied to ”Addresses_in_previous_minutes”, and the first group is emptied.

Addresses in this minute	Addresses in previous minutes
	B (13:00)

A.4 Processed minute 4 (13:01)

Now the third minute is processed and it does not contain any addresses. Due to this fact, the ”Addresses_in_this_minute” and ”Addresses_in_previous_minutes” groups do not change.

Data set			
Time	**Addresses**		
12:58	A (12:58)	B (12:58)	A (12:58)
12:59			
13:00	B (13:00)		
13:01			

Now, the second step begins and it is calculated whether any of the previous addresses has timed out.

$$13:01 - 13:00 = 1$$

$$1 < 2 \rightarrow \text{No Time out}$$

As the difference is inferior to the Time limit, ”B” does not time out and it is considered to be present in this minute. In consequence, it is copied to ”Addresses_in_this_minute” and the readings of that groups are copied to ”Detected_addresses”.

Addresses in this minute
B (13:00)

Detected addresses		
Time	**Addresses**	
12:58	A (12:58)	B (12:58)
12:59	A (12:58)	B (12:58)
13:00		B(13:00)
13:01		B(13:00)

Finally, the addresses in considered in this minute are copied to ”Addresses_in_previous_minutes”, and the first group is emptied.

Addresses in this minute	Addresses in previous minutes
	B (13:00)

A.5 Processed minute 5 (13:02)

The forth minute is processed and it does not contain any addresses:

Data set	
Time	**Addresses**
12:58	A (12:58) B (12:58) A (12:58)
12:59	
13:00	B (13:00)
13:01	
13:02	

Now, the second step begins and it is checked whether ”B” has timed out:

$$13:02 - 13:00 = 1$$

$$2 < 2 \rightarrow \text{Time out}$$

As it does, it is added to the ”Gone_addresses” group at the moment indicated by timestamp of the reading.

Gone addresses	
Time	**Addresses**
12:58	A (12:58)
12:59	
13:00	B (13:00)
13:01	

As there are no more readings to process, Step 3 begins.

A.6 Final correction

From the previous data processing, the ”New_addresses”, ”Detected_addresses” and ”Gone_addresses” groups have the following information:

New addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	
13:00	
13:01	

Gone addresses	
Time	**Addresses**
12:58	A (12:58)
12:59	
13:00	B (13:00)
13:01	

Detected addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	A (12:58) B (12:58)
13:00	B (13:00)
13:01	B (13:00)

As it can be seen, ”A” and ”B” are ”detected” even after they are gone, and therefore, this is corrected in the third step. To do so, each address saved in the ”Gone_addresses” group is processed in ”Detected_addresses” and, starting from the following minute since which the address had gone, $n-1$ minutes are traversed and deleting in them the processed address, being n the Time limit. In this example, in the case of address ”A”, the minute 12:59 is processed and in the case of ”B”, the minute 13:01 is processed.

At the end of the process, the groups will contain the following data:

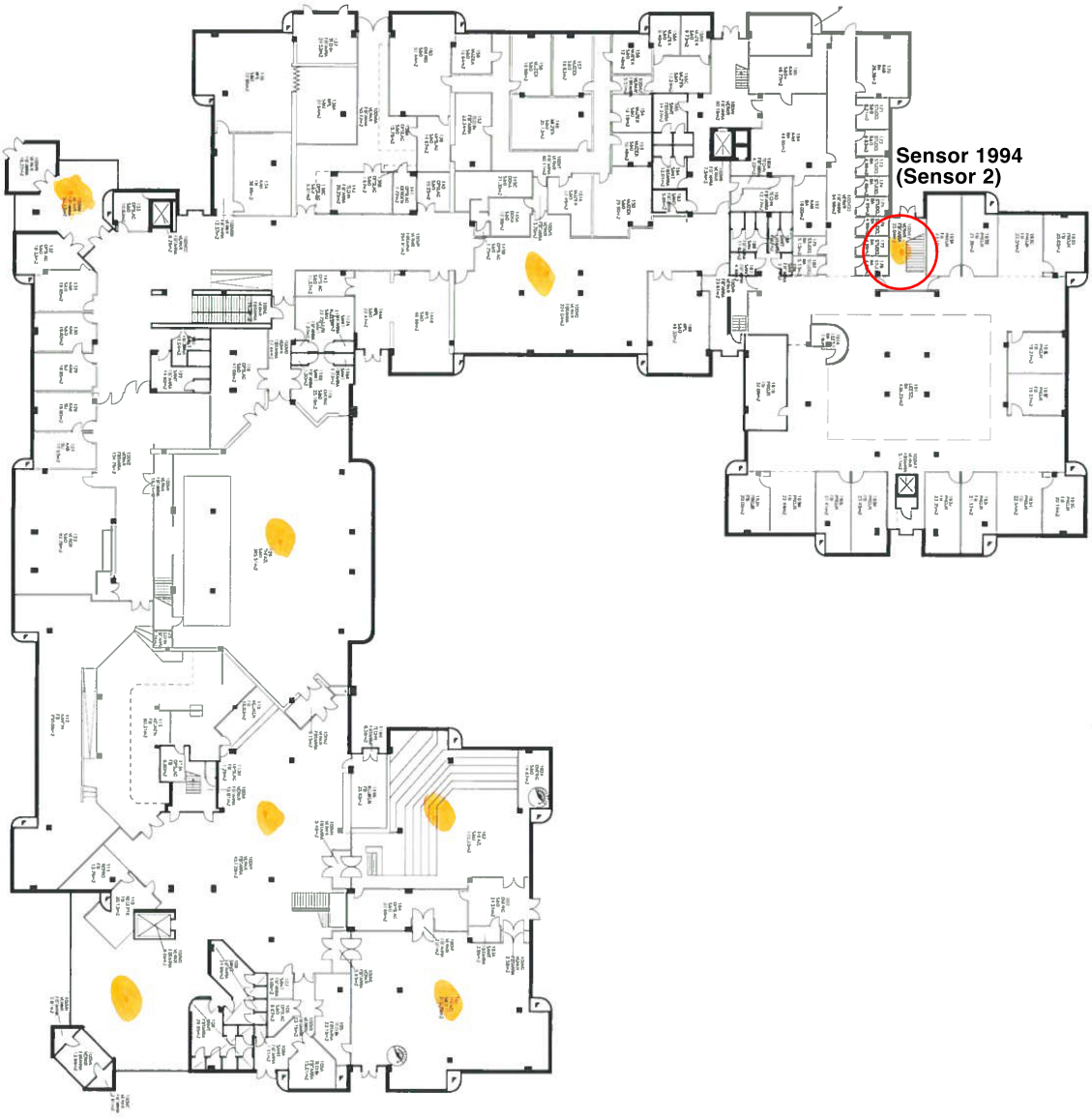
New addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	
13:00	
13:01	

Gone addresses	
Time	**Addresses**
12:58	A (12:58)
12:59	
13:00	B (13:00)
13:01	

Detected addresses	
Time	**Addresses**
12:58	A (12:58) B (12:58)
12:59	B (12:58)
13:00	B (13:00)
13:01	

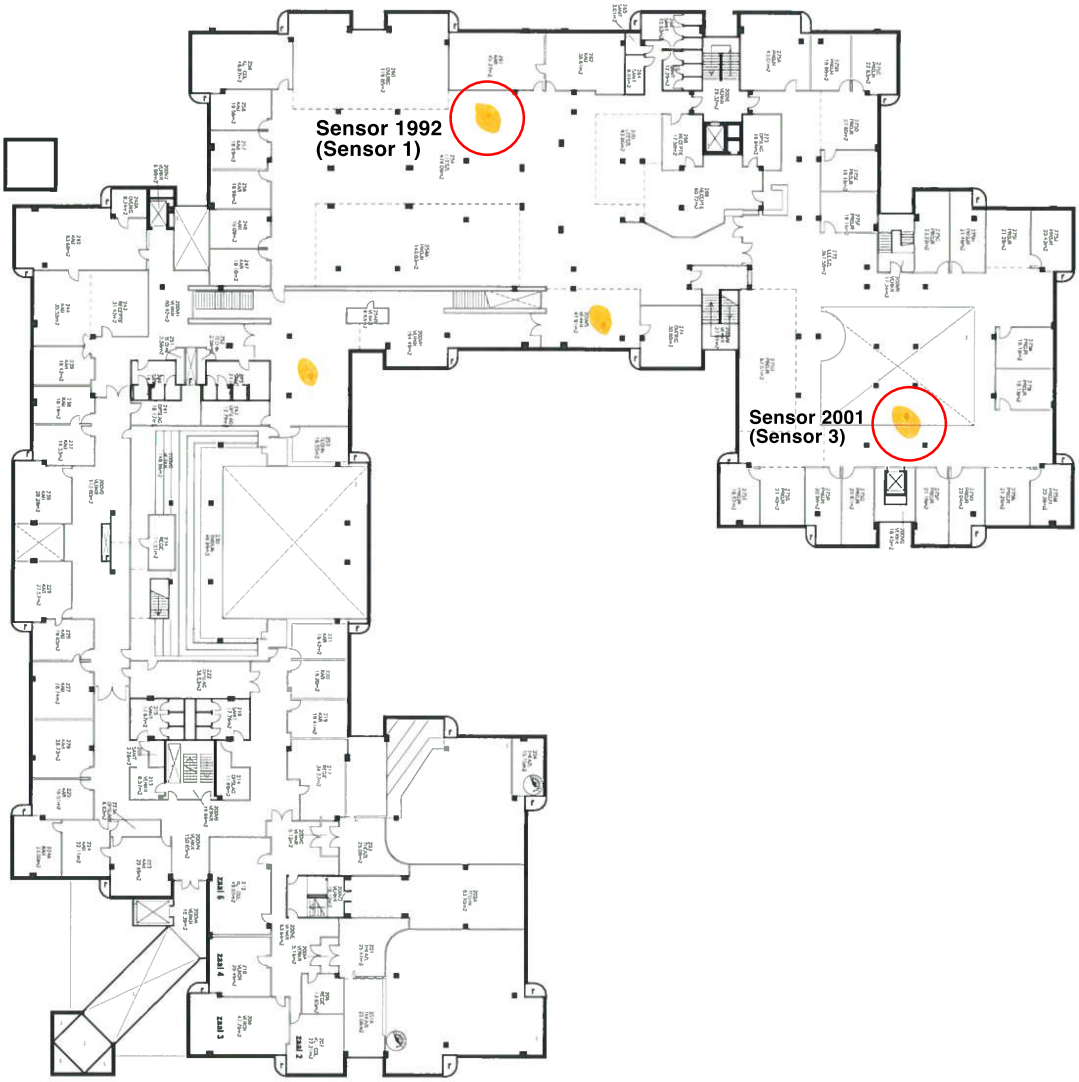
Appendix B

Map of Sensors



Gebouw : VRIJHOF
Datum : 19-03-2014
Getekend : A.S.
vloer 1

UNIVERSITEIT TWENTE.



Gebouw : VRIJHOF

Datum : 02-06-2015

Getekend : A.S.

vloer 2

UNIVERSITEIT TWENTE.