Occupancy Analyzer

Global Software Development Project

Christoffer Wilken Pagaard
Jacob Romme Rasmussen
Theresa Brandt von Fackh
Tomas Miseikis

IT University of Copenhagen

15. December 2013

Abstract

Write abstract text...

Contents

1 Introduction

1.1 Context

Smart use of energy ressources is an ongoing topic these days. The reduction of expenses is mostly the biggest impulse for companies. But also the debate around climate change brings a new legislation to reduce the waste of energy resources, whose production is damaging to the environment and future generations. The IT University of Copenhagen (ITU) has an interest in producing an occupancy model for commercial buildings, like the ITU building, to detect where energy ressources are needed and where they can be saved. Energy ressources are needed for e.g. lighting and heat-regulating systems, which are relevant for occupants in a commercial building. With the detected occupancy data the ITU can predict occupancy and develop concepts for a smart use of energy ressources in commercial buildings.

The Strathmore university in Kenya has also an interest in building up an occupancy model, but mainly for surveillance reasons. Surveillance can be used for several purposes like traffic monitoring, public safety and facilities surveillance. An IT-based surveillance system can automatically analyse the scene without the use of human ressources. By analysing the scene the detection of occupancy is a major part. Moreover a real-time prediction model on the occupancy data can be used for preventing criminal behaviour by triggering alarm or other surveillance systems.

Currently there is no existing infrastructure to build up an occupancy model in the Strathmore university or the ITU building. Both universities want a solution for an occupancy analyzer based on Raspberry PIs due to the minimal consumption of computational and monetary ressources. Furthermore the Starthmore university requests for an Android application, which represents a live-feed of the occupants in a considered room.

A group of students from both universities have to collaborate to come up with a solution for an occupancy analyzer, which can satisfy the needs of both university interests. Ideally a product should have been developed, which can be adapted to fit one or the other university needs. Furthermore a collaboration project is mandatory for the student group from ITU, in which they have to face the challenges of global collaboration, navigate compromises and come up with a solution.

This report contains the product result, design of the product, details of the project work and the learning outcomes, which were achieved in the project with the globally distributed team from the perspective of the ITU students. The project team consists of international students located in Nairobi, Kenya (East African Time) and Copenhagen, Denmark (Central European Time).

1.2 Problem

The content of this project is to build an occupancy analyzer, which detects people in a room or corridor and predicts their movement. The occupancy analyzer has to be based on Raspberry PIs and webcams, which comes with computational restrictions. A solution for the right architecture and programming languages has to be found, which can deal with those limited ressources.

Due to the usage of webcams, a visual detection of people has to be made. Analyzing images by detecting people, which are moving objects that are not part of the room, is one major challenge to face. Visual conditions of the image can change as a result of daylight. For example dynamic lighting and moving shadows should not influence the detection of people. For the best capturing of occupants and the requirement to represent the occupants on an Android application, leads to an argumentation of a reasonable placement of the webcamera. The differentiation between people, but also between people and the setting of the room, is important to detect right data about occupancy. Only if reliable data about occupancy exists, the data can be used for a reliable prediction model or can be usable for future concepts and projects.

In another step the question, how the collected data can be used, has to be considered. Building prediction models, which relies on historic data and on real-time data, is another requirement, which the project dealed with. Decissions on what kind of prediction for a meaningful application and how the data will be stored and processed have to be made.

Besides the design and implementation of an occupnacy anaylzer, another task is the collaboration of students from two different located universities. Cultural differences, difference in time, spatial distance and locally related influences have to be overcome. Different perspectives have to be combined and compromises have to be made.

1.3 Related Work

The concept of an occupancy analyzer, which uses visual recognition of occupants, is already handled in several papers and projects.

NREL IPOS Project:

1.4 Approach

1.5 Report Structure

2 | Analysis

3 Design

3.1 System Overview

Figure 3.1 displays the different components in the system and how they relate and communicate with each other.

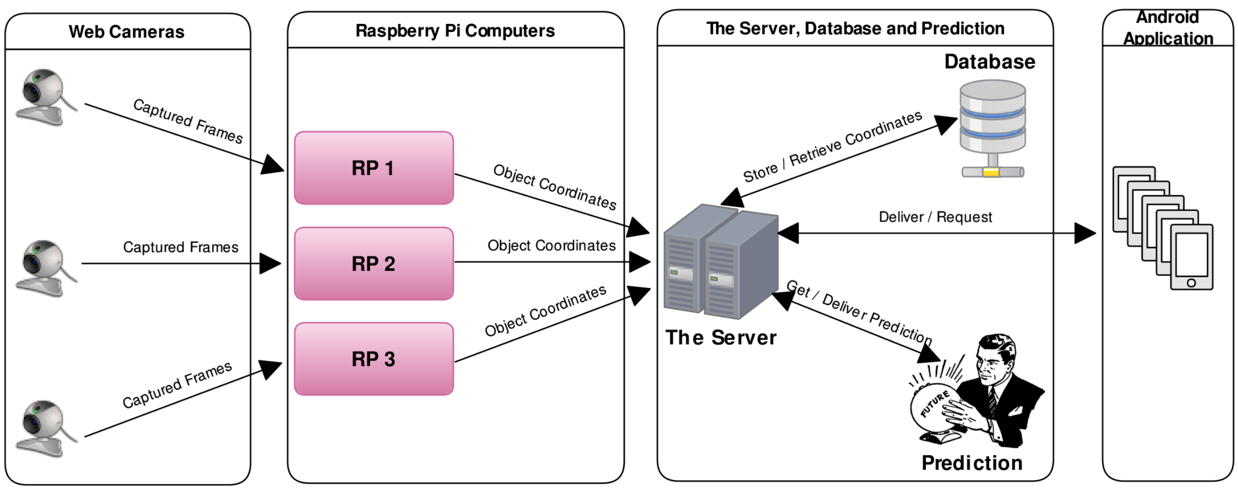


Figure 3.1: Occupancy Analyzer System Overview

As we can see, the system primarily consists of 4 components:

1. Web cameras,
2. Raspberry Pi computers,
3. The server,
4. and the Android application.

All of these components are discussed further on in the respective chapters.

3.2 Web Cameras

The purpose of the web cameras is simply to surveillance the area they have been placed in and forward the captured frames to Raspberry Pi computers for further processing, as shown in Figure 3.1. The cameras can be placed in a room, corridor, atrium or any other similar place in or outside of the building, where the services offered are required. Cameras can be placed either directly above the observed area (Figure 3.2) or in the corner of it, as illustrated in Figure 3.3. Naturally, a camera placed above the observed area would give better results, since this increases its field of view and makes it easier to correctly detect and distinguish between multiple people walking side by side. Furthermore, for the best results one must also take many different factors into account, such as the distance between the camera and monitored area, environmental conditions of the area the camera is placed in, lighting conditions.

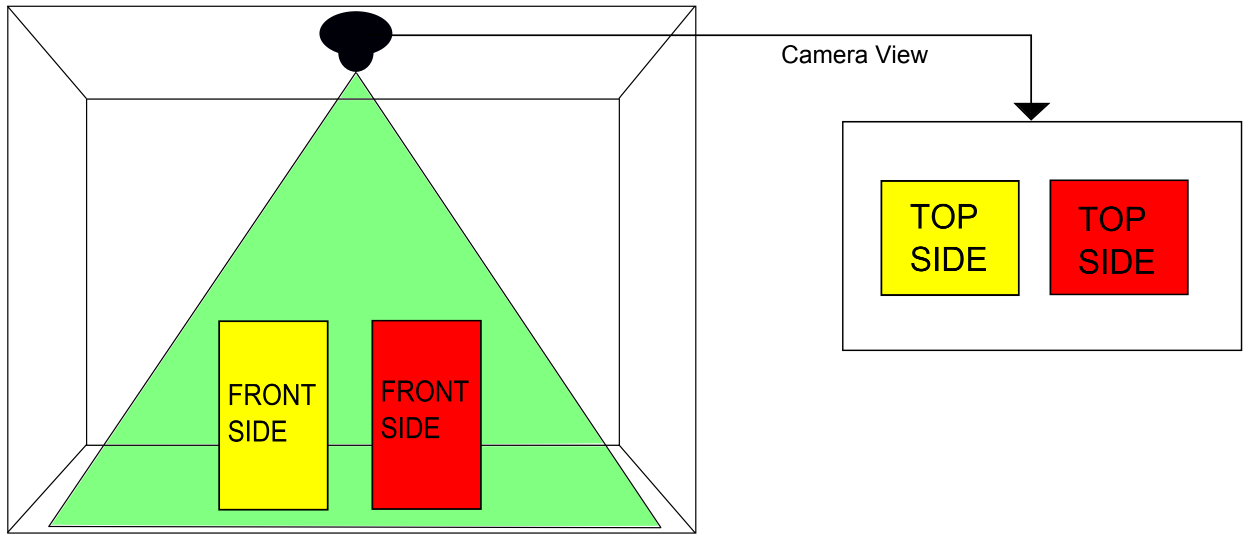


Figure 3.2: Camera top placement

3.3 Raspberry Pi Computers

The next component in the system architecture is Raspberry Pi computers. These computers have at least one web camera attached to them, and are responsible for processing the frames captured by these cameras. The main goal of processing these frames is to try to detect people in the monitored area and determine their position in that area. There are many different challenges in object detection, as well as various concepts and techniques that can be used to achieve this. We discuss these in the next sections.

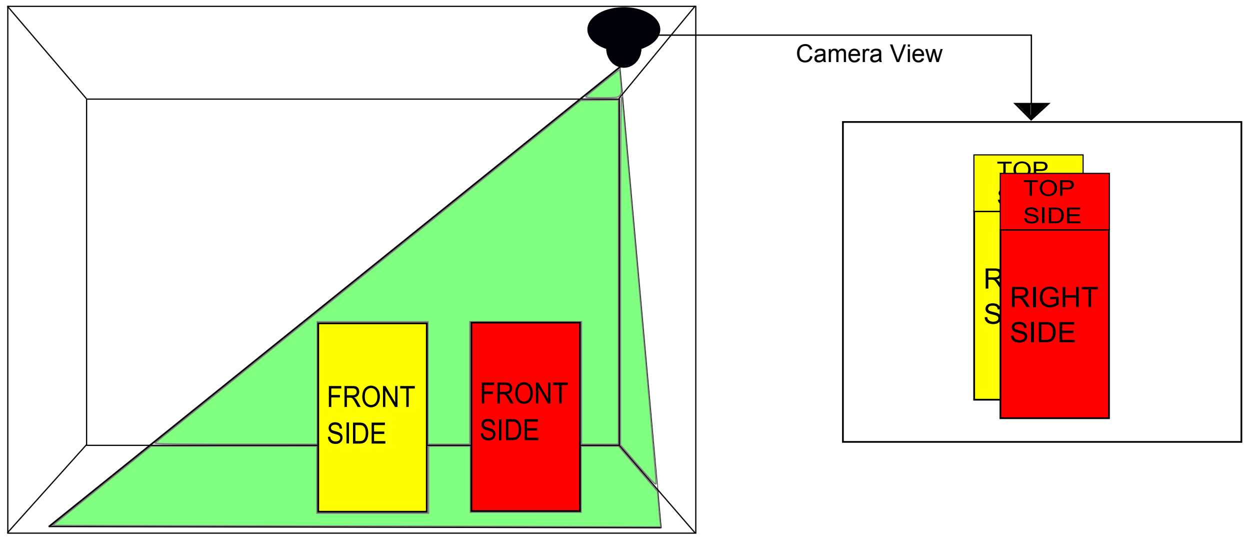


Figure 3.3: Camera corner placement

3.3.1 Object Extraction

To detect and extract objects, or in our case, people and their movement, we need to apply several motion detection techniques on the frames we are receiving from the web camera. First of all, to detect changes in some monitored area, we naturally need to have at minimum two images, which we must compare to see what changes occurred. We will try to look into two different approaches in doing this, a simple one, and one that is a bit more complicated and sophisticated, but much more adaptive and flexible.

3.3.1.1 Background Subtraction Approach

A simple approach - called Background Subtraction - would be to have a static background image of observed area that was taken prior to the analysis and did not have any people in it. Then, one can simply detect changes and movement in the area by subtracting the static background image from every newly taken image of the monitored area [1]. The difference between the two images would then allow us to see if any changes happened, since after subtraction the resulting image would either be almost totally black (Figure 3.4) - meaning no one walked passed the observed area - or the image would have some resulting bright contours of detected differences (Figure 3.5).

After doing some research and experimenting with background subtraction technique, one will quickly discover that there are multiple weaknesses to it.

- First of all, if the initial background image is always static and never changes, this technique will fail in environments where lighting is dynamic. This is perfectly illus-

Figure 3.4: Background Subtraction with no background changes

Figure 3.5: Background Subtraction with background change

trated in Figure 3.6. We can see that the lighting is much darker in the second image, possibly because the light was turned off in the monitored room, thus after subtracting our static background image from this image, the resulting image is simply a lighter version of the two images, and not the intended black image. The reason why this happens is because the original brighter background image has a much higher intensity, thus the average value of it's pixels are higher than the pixel values of the newly taken darker image. Naturally, this is a big problem, because now even if a person moves through the monitored area (Figure 3.7), he or she will not be as easily extractable as in Figure 3.5.

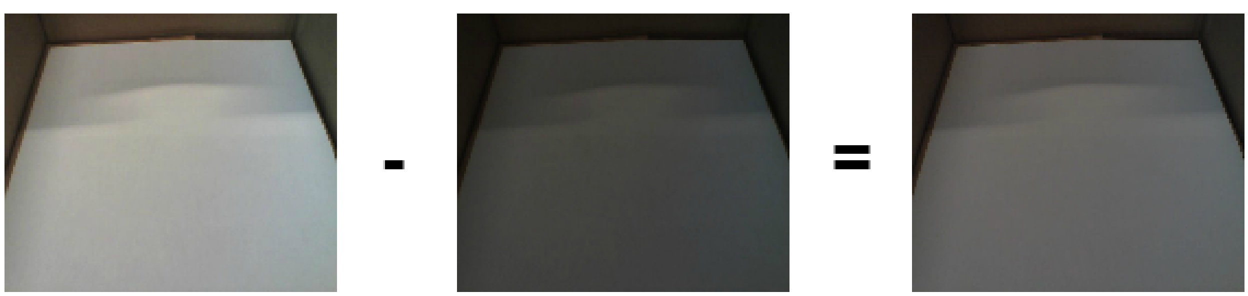


Figure 3.6: Background Subtraction with background lighting change

- Another problem with background subtraction approach surfaces when a camera is placed inside an area which has objects that constantly change their original position

Figure 3.7: Background Subtraction with background lighting change and lego figure appearing

(chairs, tables, appliances, etc.) by being moved, even small changes in object's location will spoil the resulting image after subtraction. As we can see in Figure 3.8, the object is displayed twice in the resulting image, even thought we were not even interested in it, making it much harder to detect actual people moving in the area. From now on, the resulting image after subtraction will always be corrupt unless the object is placed back in its original position.

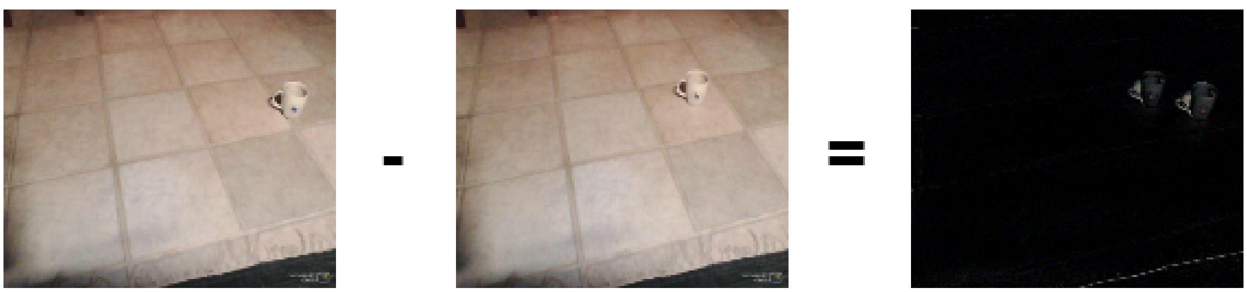
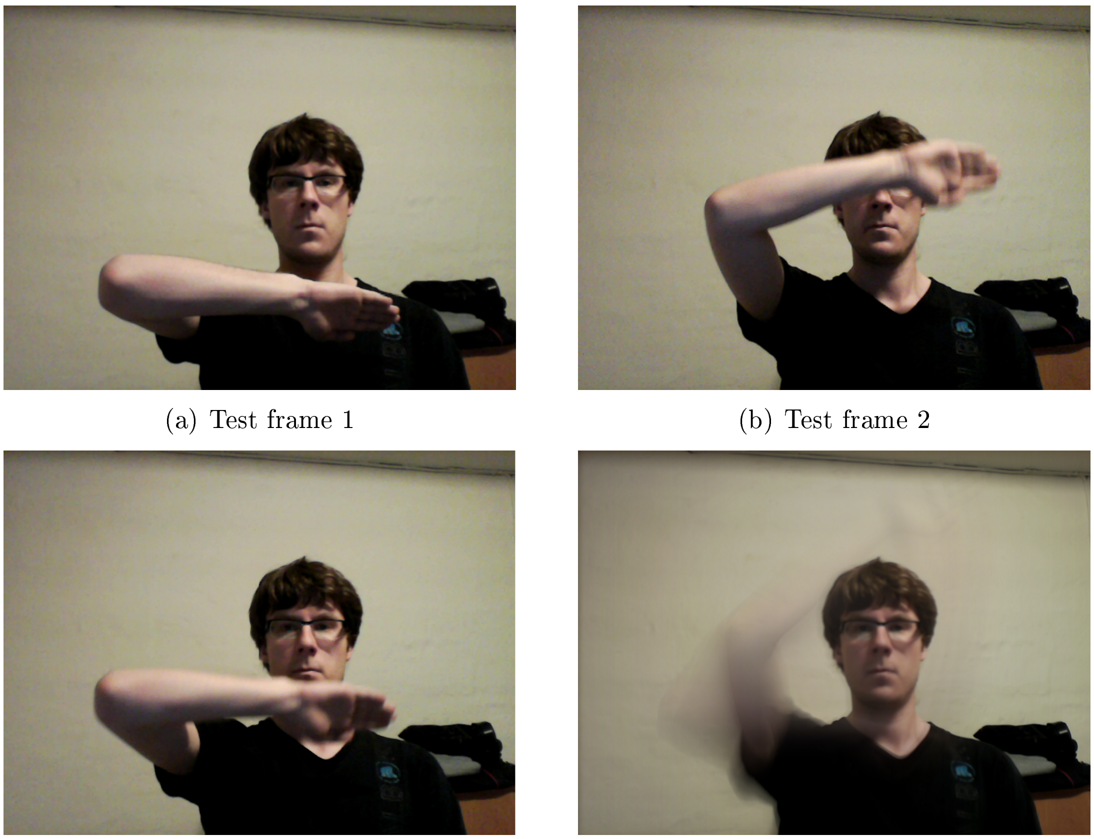


Figure 3.8: Background Subtraction with object changing it's location

In conclusion, we can see that simple background subtraction approach can work well in static environments, however it falls short in dynamic spaces. Naturally, these mentioned drawbacks of background subtraction approach need to be handled for object detection to work well, which creates additional challenges when implementing the system.

3.3.1.2 Running Average Approach

A much better approach for movement detection is using a running average method. In this technique we do not need to rely on a static background image of the monitored area taken prior to analysis. Instead, we try to find a new "approximate" background image by interpreting any changes in the background as noise and blurring them out. This is accomplished by taking a train sequence of multiple previously captured frames and performing arithmetic averaging on that sequence [2]. This exact approach is illustrated in Figure 3.9. As we can see, hand motion moving up and down (Figures 3.9(a), 3.9(b) and 3.9(c)) gets blurred out when applying running average method, thus producing an approximate background image (Figure 3.9(d)) that can be used for subtraction of the test frames from it. This method of object extraction works very well and has a lot of flexibility. It can adapt to environmental changes in the monitored area, thus eliminating most of the weaknesses that the background subtraction approach has. For these reasons, the running average approach was chosen for our design.



(a) Test frame 1

(b) Test frame 2

(c) Test frame 3

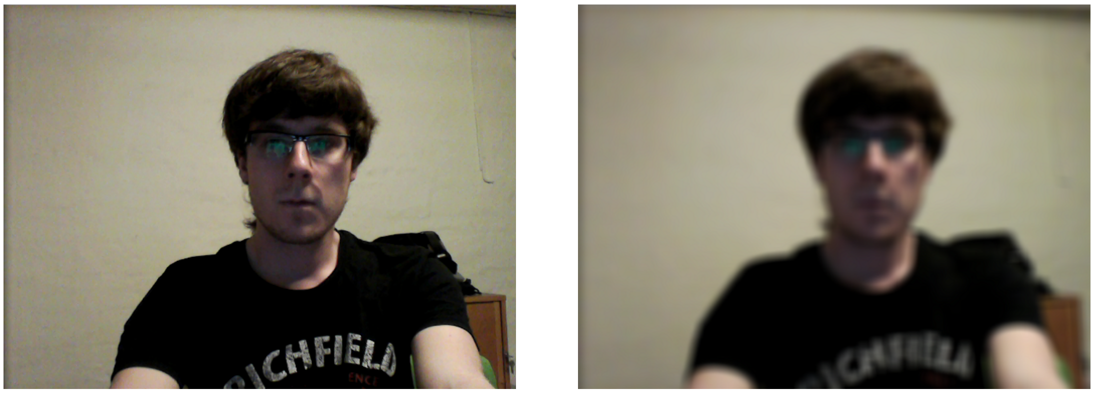
(d) Running Average frame

Figure 3.9: Example of Running Average technique

3.3.2 Object Detection

Now that we can extract objects using running average technique, we need to be able to actually find them in the resulting image we get after we perform subtraction. For this, we need to apply several key techniques in image processing.

- After we capture the initial frame of the monitored area, it will often contain noise and small details that we are not interested in. To deal with this, we must first apply blur or smoothing filter. In blurring technique we calculate weighted averages of areas of pixels in a source image by passing through it [3], which helps to reduce image noise and detail, as shown in Figure 3.10.



(a) Original frame (b) Frame after blur is applied

Figure 3.10: Blur example

- After we remove the initial noise, we can perform subtraction using running average approach (described in Section 3.3.1.2). After subtraction we will either get a nearly totally black image, meaning no motion occurred, or an image where some colors stand out, meaning some motion has occurred.
- In either case, for further processing of the taken frame, we can choose to convert it to a grayscale image. The reason for this is that the original RGB image we get has three channels, while grayscale image has only one [4], thus it is easier to work with. This procedure is illustrated in Figures 3.11(a), 3.11(b) and 3.11(c).
- For further processing of the image, we apply threshold technique, which converts the image to black and white and removes some more unwanted details and noise [5]. After threshold is applied we get the image shown in Figure 3.11(d).
- Moreover we want to expand the interesting parts of the image and contract smaller pieces, which can be considered noise and managed to slip through, even after we performed thresholding. To do so, we use two fundamental operations in morphological image processing, that is, dilation and erosion. Dilation allows us to expand the shapes contained in the image [6], whereas erosion simply shrinks shapes [7], so that bright regions surrounded by dark regions shrink in size, and dark regions surrounded by bright regions grow in size. When we apply dilation and erosion we get the image shown in Figure 3.11(e).
- Now, to project the detected area onto the original frame, we simply use bounding box technique, which gives us the coordinates of the rectangular border that fully covers the

extracted white silhouette [8] that we got in 3.11(e). Then we use these coordinates to draw a simple rectangle, as well as mark it's middle position by a red circle, as illustrated in Figure 3.11(f).

In conclusion, by applying the steps discussed in this section, we can fairly accurately detect people and their movement in the monitored area.

3.3.3 Object Differentiation

There will naturally be cases when multiple people will walk through the monitored area and will be captured by the cameras, therefore we must have a way to differentiate between them. This task becomes rather difficult if people are very close to each other, since they will simply be interpreted as one person. However, as long as people are far enough from each other, the task becomes significantly easier. There are multiple ways of differentiating between objects.

One of them is to simply look at the objects histogram, which gives a graphical representation of the intensity distribution of pixels [9]. Since people are usually dressed in different color clothes, we can simply calculate a histogram for every detected person and remember it. Now, every time we receive a new frame and detect a person in it, we go through our previously saved histograms and check whether any of them are the same as our newly detect persons histogram. If there is such histogram, we interpret the person we detected in our new frame as the same person we detected in the last, otherwise, we conclude that we have not detected this person before, thus save his histogram for future reference. The biggest weakness of this approach is that a person's clothes might have different colors from the front and back. Therefore, his histogram calculated while he is facing the camera might be rather different than the histogram of when his back was towards the camera. For this reason, if the person decides to turn around midway, he might be interpreted as a new person - never seen before by the camera - when in fact his frontal or back histogram was already saved.

Another approach of differentiating between multiple people, and in fact the approach we used in our design, is to simply use the whole frame as a coordinate system and remember the last coordinate of every single detected person. Now, similarly to histogram approach, whenever we detect a new person in the frame, we simply look throughout previously saved coordinates, and if we find that this new person's coordinates is relatively close to some previously saved person's coordinates, we simply interpret him as the same person we detected a second or or few seconds ago, otherwise we see him as a new person. Naturally, we must regularly clear our previously saved coordinates, so that newly arrived person would not simply, by taking a similar path, be interpreted as a person who is no longer in the monitored area. This approach of differentiating between people is is illustrated in Figure TBA.**TODO: add some figures with 2 or more people walking down the atrium and being differentiated.**

3.4 The Server

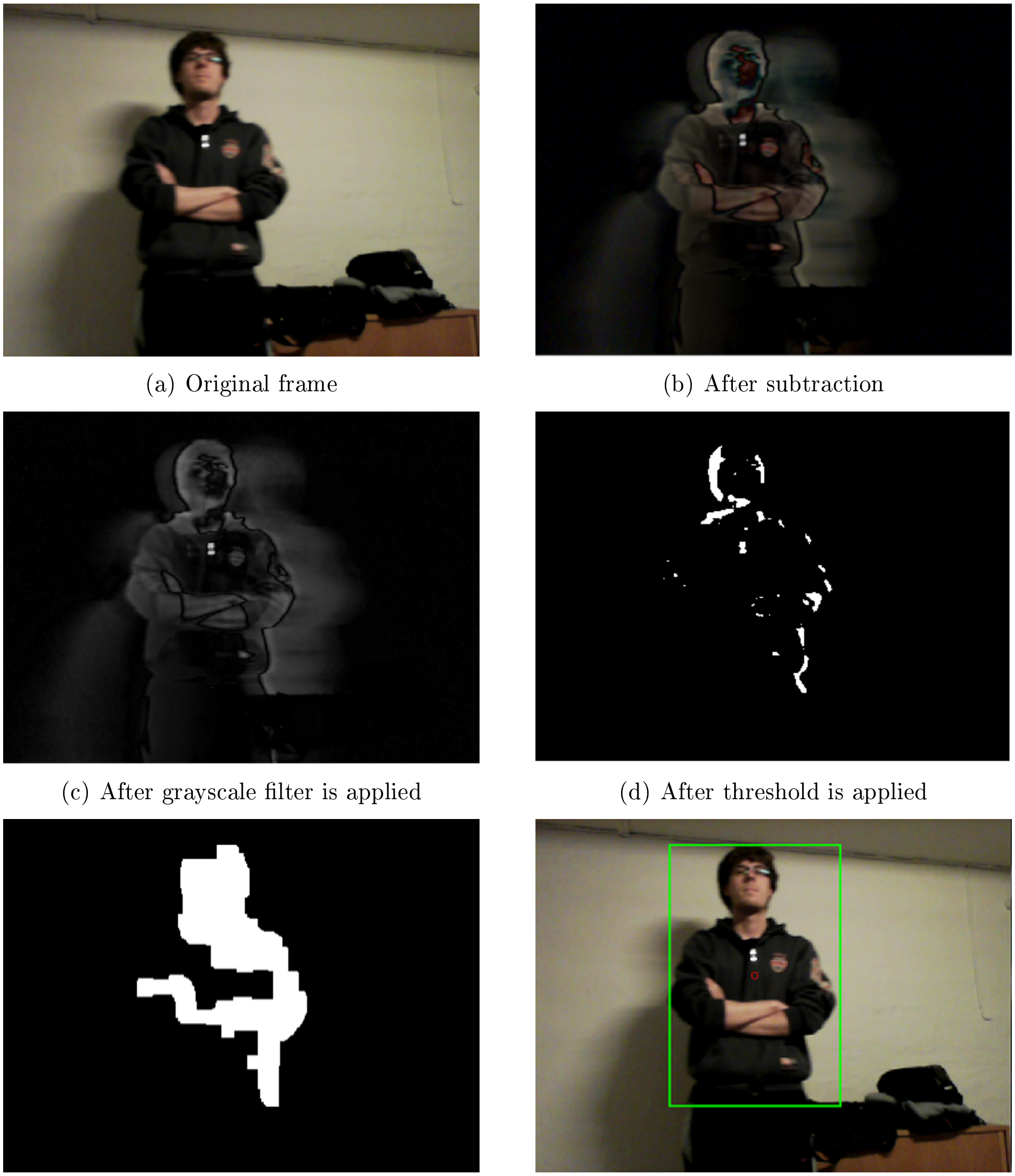
TODO

3.5 Prediction

TODO

3.6 Android Application

TODO



(a) Original frame

(b) After subtraction

(c) After grayscale filter is applied

(d) After threshold is applied

(e) After dilation and erosion

(f) After bounding box is drawn

Figure 3.11: Object detection example

4 | Implementation

5 | Evaluation

5.1 Verification

5.2 Benchmarks

6 | Collaboration

6.1 Circumstances

- Change of the project

6.2 Projectmanagement Method

- Classic approach
- SCRUM method
- Our method (Structure)
- Weekly meetings internal, global, with advisor
- Time recording
- Splitting up assignmnts
- Plans

6.3 Project Team and Organisation

- Project team members
- Skills
- Preferences
- Roles
- Splitting of the technical parts

6.4 Timeline

- Overview of the phases, milestones, deadlines, other exams/hand-ins
- Ressource planning
- Time-Recording (Toggl)

6.5 Issues

- Illusion of the project work and project team
- Failure to comply with the assignments
- Communication
- Lack of skills
- Other exams/hand-ins
- Differing requirements
- Attendence of meetings
- Equipment

6.6 Hypothetical Scenarios

- Assignments
- Communication
- Requirements
- Organisation by the universities (Requirments, clarification,)

7 | Discussion

8 | Conclusion

9 References

[1] Massimo Piccardi. 2004. *Background subtraction techniques: a review*, p. 1.

[2] Mohamad Hoseyn Sigari, Naser Mozayani and Hamid Reza Pourreza. February 2008. *Fuzzy Running Average and Fuzzy Background Subtraction: Concepts and Application*, p. 1.

[3] Herman Cheng, Zhicong Huang and Mark Kumimoto. Spring 2006. *Final Project Report - Image Processing Techniques*, pp. 2-4.

[4] Matt Siber. Fall 2005. *Resolution and File Size*, p. 1.

[5] Brian Lovell *Thresholding Techniques*. Spring 2003.

[6] R. Fisher, S. Perkins, A. Walker and E. Wolfart. 2003. *Dilation*.

[7] R. Fisher, S. Perkins, A. Walker and E. Wolfart. 2003. *Erosion*.

[8] Dr. Nicolas Pronost. *Introduction to Image Processing*, p. 9.

[9] Ahmed Elgammal. Spring 2008. *Histograms of Digital Images*.