Augmented Reality in Surveillance Systems
- a concept study at Axis Communications AB

Jonas Nolte & Magnus Wetterberg

Master's Thesis

Department of Design Sciences
Lund University
ISRN:LUTMDN/TMAT-5170-SE

EAT 2013

“Fiction is not imagination. It is what anticipates imagination by giving it the form of reality. This is quite opposite to our own natural tendency which is to anticipate reality by imagining it, or to flee from it by idealizing it. That is why we shall never inhabit true fiction; we are condemned to the imaginary and nostalgia for the future.“

Jean Baudrillard

Abstract

Augmented Reality in Surveillance Systems

by Jonas Nolte and Magnus Wetterberg

The aim with this thesis is to investigate the possibilities to use Augmented Reality (AR) in surveillance systems. The thesis will examine ideas suited for network based cameras but also other devices such as a head mounted display (HMD). AR today is mainly used in smartphones as entertainment and therefore the approach to use AR in surveillance systems will cover new areas.

The project will be divided into several steps including investigation, design, tool-evaluation and implementation. Investigation includes numerous brainstorming sessions and research along with storyboards and scenarios. The investigation will lead to a product to be designed and implemented in a selected platform. To design the application, a Lo-Fi and Hi-Fi prototype is created.

As a result, a final application was implemented on the platform Android which uses Google Maps as an overview of the camera system. All the cameras are viewed in different manners and in the image, the enhanced AR information is applied to increase the information output and orientation. Axis Communications network cameras where used in the application.

Sammanfattning

Augmented Reality i ett övervakningssystem

av Jonas Nolte och Magnus Wetterberg

Syftet med denna uppsats är att undersöka möjligheterna att använda Augmented Reality i övervakningssystem. Arbetet kommer att utforska idéer anpassade för nätverkskameror men även andra anordningar såsom en head mounted display. Idag används AR huvudsakligen i smartphones som underhållning, vilket innebär att det här projektet kommer leda in AR till nya områden inom övervakning.

Projektet är uppdelat i flera steg: utredning, design, utvärdering av verktyg och till sist implementering. Utredningen innehöll brainstormingsessioner och forskning som tillsammans med storyboards och användarfall gav en helhet. Till sist definierades en produkt som skulle utformas och genomföras på en utvald plattform. Vid designfasen skapades en Lo-Fi-prototyp för att undersöka om testpersonerna använde systemet som avsett. Lo-Fi- prototypen gav även ytterligare feedback till ett bättre system.

Som ett resultat av alla steg har en slutlig applikation skapats till plattformen Android. Applikationen använder Google Maps för att för att skapa en översikt i kamerasystemet. Bilderna som kamerorna visar är förstärkta med AR, vilket förbättrar bildinformationen samt ger en bättre orientering än tidigare.

Acknowledgements

We would like to thank our supervisor Mats Bylander at Axis Communications who gave us the opportunity to carry out this master thesis. A huge thank to Joakim Eriksson, our supervisor at LTH for taking part of the project and who came up with a lot of useful and good feedback during the project development. Also thanks to all of the other Axis employees that contributed with insightful and important thoughts of the project and of all our different ideas that was carried out during the project.

Jonas Nolte & Magnus Wetterberg
Lund, June 2013

Contents

Abbreviations

AR	**A**ugmented **R**eality
POI	**P**oint **O**f **I**ntrest
GPS	**G**lobal **P**ositioning **S**ystem
RGB	**R**ed **G**reen **B**lue
PTZ	**P**an **T**ilt **Z**oom
IR	**I**nfra**R**ed
HMD	**H**ead **M**ounted **D**evice
IP	**I**nternet **P**rotocol
API	**A**pplication **P**rogramming **I**nterface
SDK	**S**oftware **D**evelopment **K**it

Chapter 1

Introduction

1.1 Background

The main idea of this master thesis began during the summer of 2012. The work and the research had the objective of looking at the different possibilities to create an image fusion between images from an Infrared camera combined with images from a standard RGB-camera. An RGB-camera that presents the world as we see it and the infrared camera that captures the heat radiation from an object. The image stream from the cameras was combined in different ways to increase the image output so the image output could present more information. More about image fusion in section 1.4. Another goal when creating an image fusion was to improve the orientation of the scene which is very important in surveillance. Thoughts of an alternative way to further increase the orientation and the image information founded the idea to use AR. More about AR in Section 1.3.

1.2 Axis Communications AB

Axis Communications was founded in 1984 by Martin Gren and Mikael Karlsson in Lund, Sweden [1]. Axis Communications started out developing protocol converters and printer interfaces. In 1996 the first IP-based camera in the industry was introduced. Axis Communications still has its headquarters in Lund and is today a market leader providing and developing network based camera solutions. Axis Communications is always in the forefront of new camera solutions and camera techniques.

1.3 Augmented Reality

Augmented means that something has been made greater in size or value, and in the case of AR it gives changes in a way that affects our view of real-world environment. One way of doing so is to mix reality with graphics. AR is used in real-time applications to get the immediate effect of a change in the real world. AR can be of assistance for orientation and learning in an environment. This can be an environment where it can be hard to find out the location of a device in a system or when a device has a non-logical name and where the user needs practice to understand and learn how the system works. AR is not only limited to the sense of sight and can be applied to senses such as touch, smell or hearing. In this thesis only the visual sense of AR will be treated.

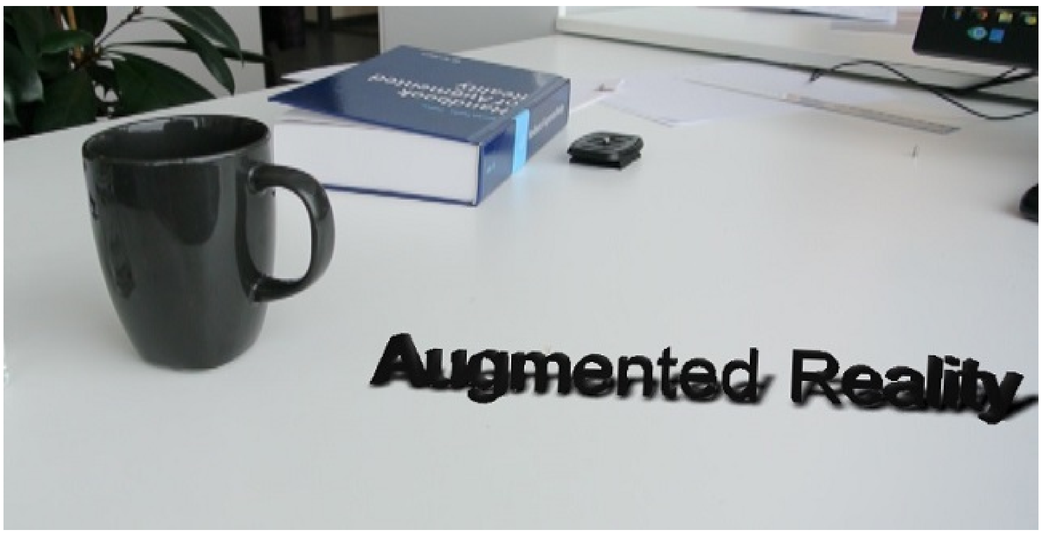


FIGURE 1.1: The real world augmented with text information.

1.4 Image Fusion

The Image fusion technique uses information from two or more image sensors, often information from a sensor in the infrared spectrum and one from the visual spectrum. The visual spectrum is captured with an RGB-camera and the Infrared camera registers electromagnetic radiation of infrared waves and gives us a powerful way to see things that are hidden within our own visual interpretation of nature. The image from the sensors is then combined and creates the output image. The resulting image will be more informative than the input images, as shown in Figure 1.2 where two images are combined, one from the infrared camera and one from the camera that is working in the visual spectrum.

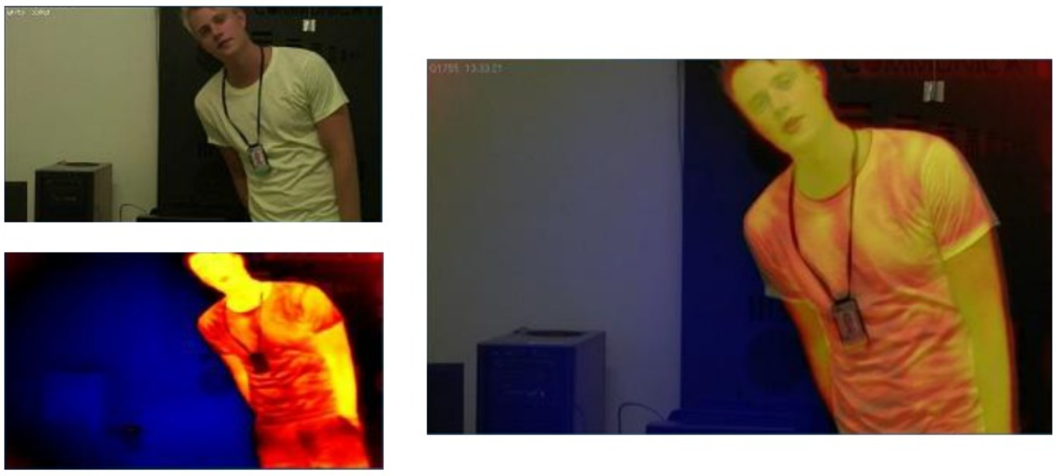


FIGURE 1.2: Infrared input image at the top left, RGB input image at the lower left. Image fusion to the right.

1.5 Aim and Purpose

The aim with this thesis is to investigate the possibilities to use AR in surveillance systems. By exploring and demonstrating the benefits of AR, a final application using Axis cameras will be implemented. Before an application can be developed, it's important to examine existing AR-applications and their possible contributions.

The thesis vill focus on the use of various cameras from Axis Communications. Most of the focus will be to enhance the image information on the individual camera but also focus on letting multiple cameras communicate in a greater system.

The purpose of adding graphical content is to increase the orientation and enhance the image with more information. Improved orientation can also be important during bad weather conditions such as fog, snow and limitations of light.

1.6 Limitations

There are a few limitations to this project. One limitation is the cameras that are connected to the application needs to be on the same network as the applications device. If a tablet or smartphone is used they need a wireless network connection and therefore limited amount of cameras can be mapped to the same network. In the Axis building only one camera is connected to the wireless network and only one is allowed on the wireless network due to the network configuration and network setup in the building.

It would also be desirable to implement the AR- part directly on the camera. Unfortunately there are no GPU[1][2] in the cameras and therefore no support of OpenGL[2][3]. It would also consume too much time to implement and run any AR application on a GPU, due to the complexity and difficulty. Therefore a separate device will retrieve an image stream from the camera. The received camera stream can later be analyzed and displayed on a device. The device will either be a computer, tablet or a smartphone.

[1]GPU: "a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display." Reference: `wikipedia.org/wiki/Graphics_processing_unit`

[2]OpenGL: "a cross-language, multi-platform API for rendering 2D and 3D computer graphics." Reference: `wikipedia.org/wiki/OpenGL`

Chapter 2

Technology Background

2.1 History

The history of AR goes back to the $1950's$. In the $1950's$ the first idea by Morton Heilig came up where he constructed a cinematic show that activated more senses of the viewer [4]. In 1962 the first prototype named "Sensorama" was implemented by Morton and this famous AR-machine was the first step in the AR-evolution, seen in figure 2.1. Later on in 1966, Ivan Sutherland invented the first head mounted display (HMD, discussed in Section 2.3.1) [4].

Since the first inventions and implementations the AR-technique has been developed regularly to this day and has in the latest years grown tremendously thanks to the smartphone industry. The smartphone has many sensors and has the ability to see the world through an "eye", the built in camera. By combining the sensors with the camera an advanced AR-tool can be implemented [4].

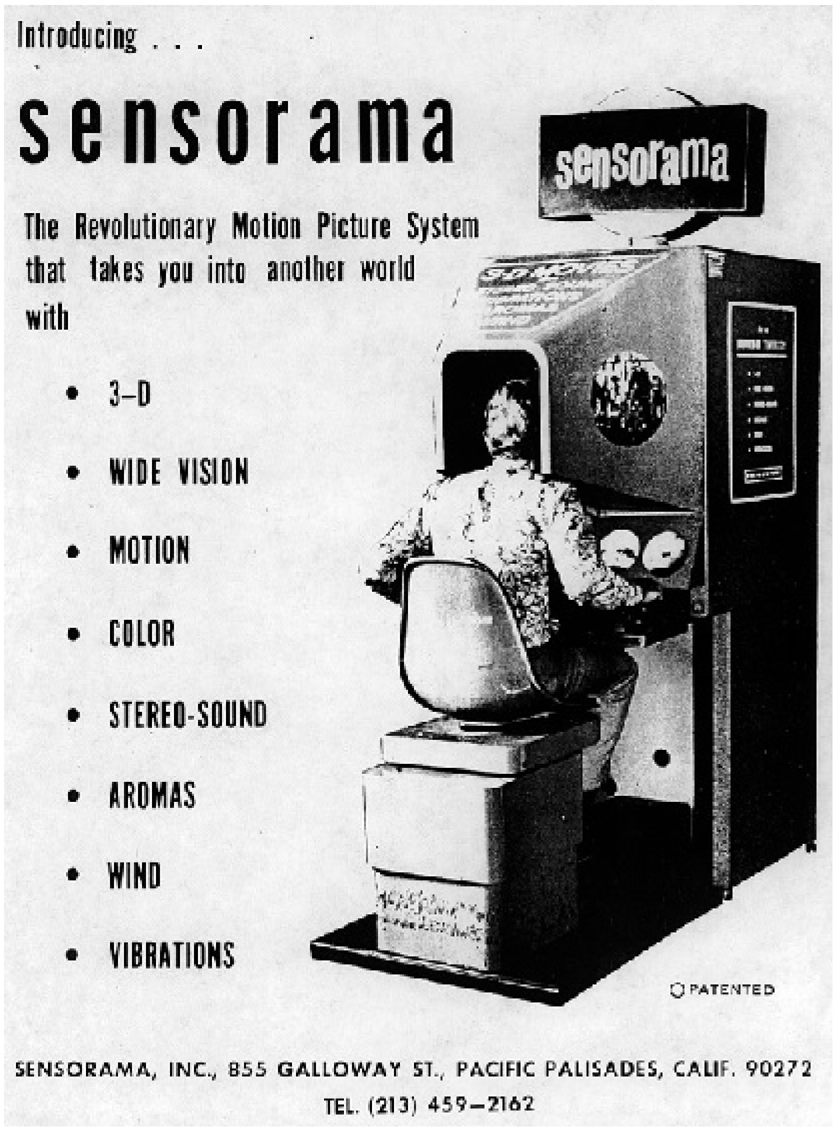

Introducing . . .
sensorama
The Revolutionary Motion Picture System that takes you into another world with
• 3-D
• WIDE VISION
• MOTION
• COLOR
• STEREO-SOUND
• AROMAS
• WIND
• VIBRATIONS
sensorama
PATENTED
SENSORAMA, INC., 855 GALLOWAY ST., PACIFIC PALISADES, CALIF. 90272
TEL. (213) 459-2162

FIGURE 2.1: Sensorama by Morton Heilig, Pioneer in AR.

2.2 Related Work

AR has been investigated for quite some time and a lot of areas and industries have been covered with ideas and applications. Many master thesis has been made and some of them at LTH, such as *Augmented Reality Visor Concept* [5] and *Design and Implementation of a Mobile Application for Public Transport* [6]. Even though the thesis's are well made, they tend to focus on either entertainment, transportation or tourism [4].

Besides the master thesis, companies around the world have created plenty of applications. *Wikitude* [7] is an AR browser which lets the user sort by different categories and point out targeted objects. The location-based tracking uses the camera on the phone together with its sensors and places a billboard in the correct direction to the object. Figure 2.2 shows the AR part of the application.

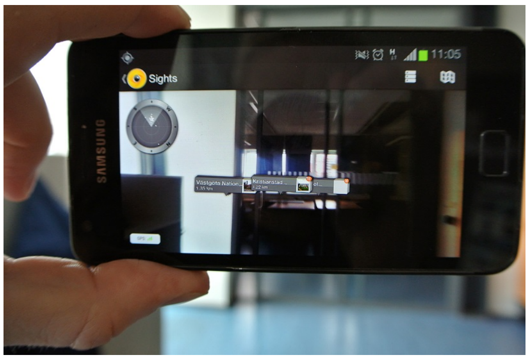


FIGURE 2.2: The AR application Wikitude.

Another way to use AR is to analyze the image with face recognition and let the user try out some products. Ray-ban has done this and it's possible to try their entire range of Ray-ban glasses right from the computer with a simple webcam [8].

These are just a few examples on some applications and there are plenty of different articles, apps and concepts out there. Even though the area is examined, it does not include the field of surveillance where AR is yet unexplored.

2.3 Displays

The choice of which displays to use in AR are very important and the need for different displays varies with different situations and usage areas. From the surveillance and the observer point of view different techniques will be discussed in this chapter such as head mounted displays (HMD), computer screens and smartphones.

The reason for treating the subject of HMDs is due to the powerful use in the AR field. There are a lot of upcoming ideas that is based on this kind of display. Computer screens are addressed in the text due to its advantage on the market. The displays of smartphones are discussed because of the smartphones importance in the AR field.

2.3.1 Head Mounted Displays

HMD is not a new technique and has been known for quite some time now, even though it's not until this century HMD really has taken its form. At the beginning HMD was very heavy and uncomfortable. One of the first persons who started the research about HMD was the pioneer Steve Mann (Figure 2.3) [9].

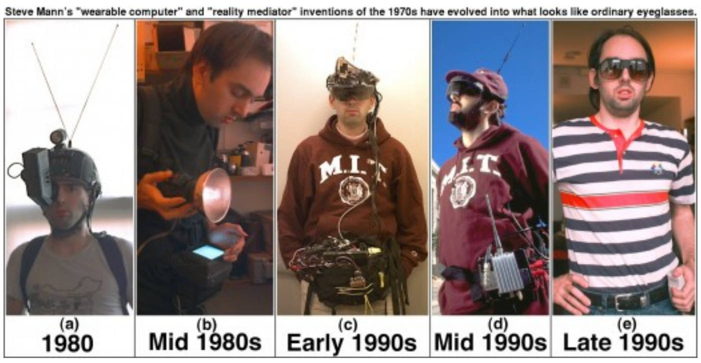

Steve Mann's "wearable computer" and "reality mediator" inventions of the 1970s have evolved into what looks like ordinary eyeglasses.
(a) 1980
(b) Mid 1980s
(c) Early 1990s
(d) Mid 1990s
(e) Late 1990s

FIGURE 2.3: Steve Mann with the evolution of HMDs.

The evolution has come a long way since that day and many manufacturers are constantly releasing new hardware. Google will in 2013 release their latest HMD which is called Project Glass and has been compared with Steve Mann's prototype from 1999 (Figure 2.4) [10].

There are two different kinds of HMDs, optical see-through and video see-through devices. The optical variant lets the user see the real world with their own eyes and projects

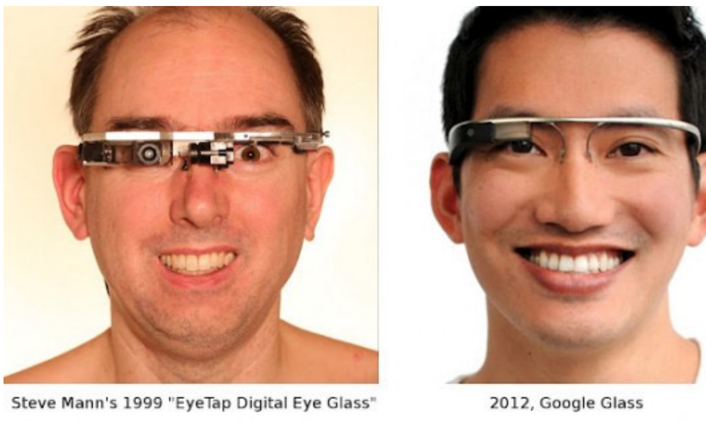


FIGURE 2.4: Steve Manns HMD comapred with Googles new Project Glass.

the graphics from a monitor into the user's eyes. This brings a great advantage because everything the user sees is in real-time and appears natural for the human eye. The view is also not restricted by a cameras resolution as in the case of a video see-through device and if the power is shut off, the user can still see what happens. Still be able to look through the device without power could be a matter of safetey.

The optical see-through lacks an important feature that video see-through has. It doesn't have a video camera. The information from the camera can be interpreted and the graphics can be adjusted to the environment. Video see-through uses this information and displays it on a monitor which the users looks at directly. If the power goes off, the user will be completely blind and accidents may happen.

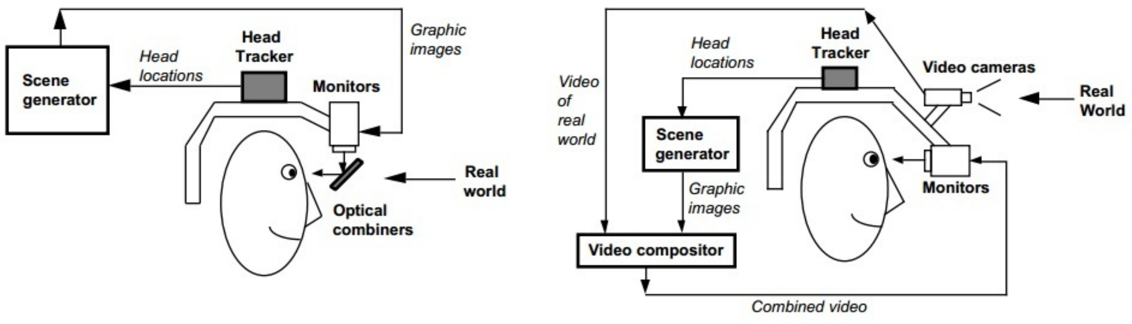


FIGURE 2.5: Left: Optical See-Through, Right: Video See-Through.

Both devices have their advantages and disadvantages and which of them that suits best depends on the different situations and applications. Therefore different devices may differ from the scenarios described in this report in the area of AR [11].

2.3.2 Computer Screen

Computer monitors has been used ever since the first home computer was released in the late 1970's. Most people know how to use them and what they can expect from it. Even though the monitor is commonly used the screen lacks many features. The field of view which increases the user's presence and impression is limited and the fact that the screen is not quite mobile makes it hard to use in the field of AR [12]. It can, however, be used where presence doesn't matter. If the goal is just to improve the information from the video input computer screens can be most useful and since most users already own a computer screen there is no need for extra costs and configurations. The monitors resolution is an important feature and since more than 85% of the users have higher resolution then 1024x768 (2011) it's no longer a drawback [13].

Figure 2.6: A 3D Full-HD computer monitor from LG.

It's also becoming more commonly that monitors support 3D experiences with separate glasses and that monitors uses a touch display technique which is a possible choice of the interaction method. A higher presence and immersion can be achieved with 3D, high resolution and larger screens. The monitors can be a good choice for AR in today's usage but will fade away when HMD gets cheaper and covers a greater usage area.

2.3.3 Smartphones

Plenty of what is going on in the market today is developed for smartphones. Many developers of SDKs[1] is working with Android or iPhone[4]. This is due to the phones capacity of built in sensors like GPS, compass, gyro and the camera. With this senses the phone becomes a powerful tool for AR-applications. It is possible to know from the GPS position where the phone is located in the world. The compass gives you the direction, gyro provides information such as if the phone is tilted in any direction and the camera prolongs the vision of the human eye. The camera is one of the strongest parts when working with AR and the reason is the visual feedback it can give. Graphics that is being added and information in real time on the screen gives the user a whole new experience.

2.4 Interaction Methods

Gestures, *Touch* and *Natural language* are the three most used interaction methods. They represent most of the movements and gestures one can do when interacting with a system.

- *Gestures* is one of the most basic interaction methods for humans [14]. This because gestures are simple and that they are the most undelaying interaction to interpret between humans. This goes for interaction between system and human as well. Gesture is the natural way of swiping between pages on a touchscreen or when interacting with a system that recognize the whole body language, such as moving the arms up and down or the torso of the body.

- *Touch* is the interaction technique where interaction is mapped to movements such as touching the screen in different manners. Pinching, scrolling or detection of multiple finger movements are some of the gestures that can be used during interaction. Using touch during interaction makes the interaction fast and simple

[1]"a SDK is typically a set of software development tools that allows for the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform." Reference: `wikipedia.org/wiki/Software_development_kit`

and the user gets feedback immediately because of the visibility on the screen [4].

- *Natural Language Interface* map reality with interactions of speech [15]. Talking or to give simple instructions to a system can be powerful because no keyboard or touchscreen is required. The drawback can be to get started using the system. Learning process takes some time and the feedback from the system while giving input can be difficulty to get if there is no screen or other visual feedback.

2.5 Tracking

Tracking is used to determine where to place the 3D object in the image. The different tracking methods offer varying levels of accuracy and precision and require different computational power.

Marker-based tracking is techniques that tracks or recognize a certain marker that has a specific pattern, such as a QR-code [4]. The idea is to find and analyze the markers in the image and get the coordinates and the plane information from the marker. The coordinates and plane info is needed to be able to calculate the normal and the coordinates of the plane so 2D/3D models or other information can be presented on the marker.

Another tracking method is *marker-less tracking* [4]. Instead of using a marker for determine the objects position, other techniques are used. One method is to use mathematical algorithms on the image to find desired patterns such as faces, roads or buildings. Since the image is being analyzed the technique is complex and requires many computations. Instead marker-less tracking can rely entirely on sensor inputs. With the GPS-position, gyro meter and the compass it's possible to place the 3D-object on the correct position. Without the image analysis this technique has low complexity and is feasible to implement on an Axis camera [16].

2.6 Camera

The cameras of Axis Communications is based on communication over the Internet Protocol (IP)[2] and the images are available over the network as an image stream. To be able to receive an image the application needs to be connected to a network that has access to the camera. The video stream can then be received as a Motion-JPEG stream or via image reads over HTTP[3].

The work will consist of trying to work out the graphical use and solutions of mainly two different camera constructions. One of the cameras is the static camera (Figure 2.7) where no movement is applied but where the direction of the camera and field of view can be of interest. The other camera is the (Pan-Tilt-Zoom) PTZ-camera as can be seen in Figure 2.7. The PTZ-camera can rotate, tilt and zoom so this kind of camera can be in most interest for an AR implementation.

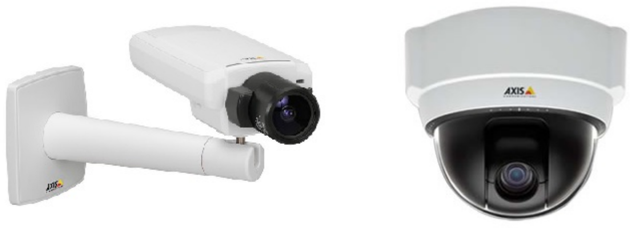


Figure 2.7: Standard Axis camera to the left. Axis PTZ camera to the right.

Cameras in the *visual region* dominates the world. This is because they are working in the visual region of the spectra, the spectra of our own visual perception. Working in this field is often enough to get the important information in an image.

Cameras in the *infrared region* are different from the ordinary visual camera. The benefit of using a infrared camera is that the camera works equally well during nighttime as in daytime. Since the camera is sensitive for heat radiation the infrared camera often called a thermal camera picks up radiation from warm objects like humans. So it is now very efficient to detect someone walking around in a forbidden area in total darkness or

[2]”IP is the principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet.” Reference: `wikipedia.org/wiki/Internet_Protocol`

[3]”HTTP is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web.” Reference: `wikipedia.org/wiki/Hypertext_Transfer_Protocol`

simply detect something during a sunny day. Example of a typical thermal camera is shown in Figure 2.8 [17].

Figure 2.8: Product of Axis - Thermal Camera.

Chapter 3

Concepts

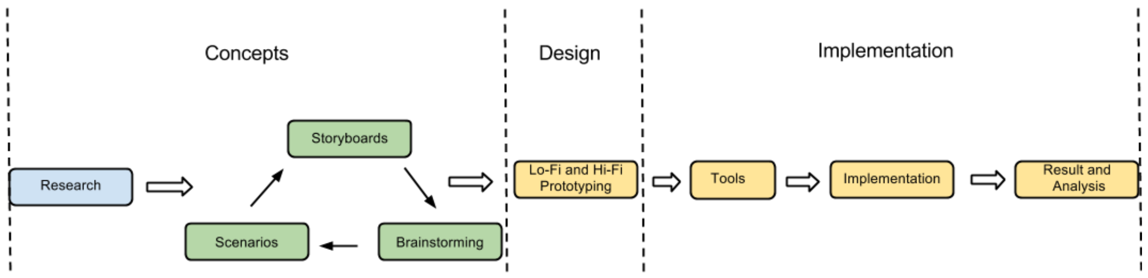


FIGURE 3.1: Timeline of the project.

This chapter will go through the different steps of the conceptual phase. From the beginning were only ideas existed, to the definition of the actual product with the design and implementation phase involved. Figure 3.1 gives an overview of the whole project to get an idea of the different phases.

The result of the brainstorming sessions, scenarios and storyboard led to the different ideas that is visualized in Section 3.5. To reach a conclusion of what to implement in the final application, an investigation was done by working iterative with the visualizations.

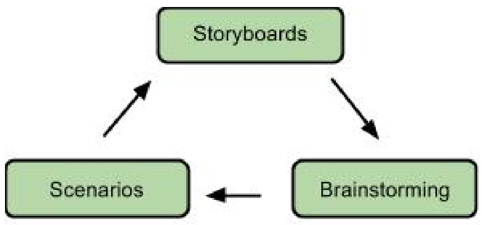


FIGURE 3.2: Iterative approach to create conceptual visualizations.

3.1 Research and litterature review

The area surrounding AR is constantly updated and the latest research is available on the Internet. The research and literature review during the project was a result of looking at different implementations and reading of earlier projects online. Since many publish their work on YouTube, this was one of the greatest sources for the latest AR applications. Most of the information came from the Internet but the book *Handbook of Augmented Reality* [4] also gave some input in the area.
When the base of the project took form, a lot of time were spent looking at all of the different AR techniques and the different implementations with their scenarios. One important thing doing so was to get inspiration for the brainstorming and bodystorming sessions.

3.2 Scenarios

To get an insight of the problem, scenarios were created. These were used in a creative way to come up with different ideas and to define the difficulties with today's system and method of use. As a background for the scenarios and to imagine the different situations, the camera system which is used today was analyzed. Also an amount of images taken by the cameras was collected to see what could be improved.

- **Scenario 1**
  "You are an operator of a surveillance company which have control over hundreds of cameras. It is your duty to make sure nothing gets unseen and that incident gets reported or alarmed. Even though many cameras are your responsibility only a few monitors can view its content. In one of your monitors something suspicious is going on. There is a man that walks in a restricted area without permission, even though you can see him the camera is far away and has no clear image of the persons face. The person that is under suspicion starts running and gets too far away from the camera and you need to change the camera, unfortunately the surrounding cameras are not one of the one monitors which you can view and you need to change to the specific camera which you know is located where the person

got out of the image. Because of the hundreds of cameras which is in your command you didn't have time to change camera and the person walks away uncaught"

- **Scenario 2**
  "As an operator of a surveillance company, the shifts can be long and at late hours in the night. The monitors that you view often displays the same scene and nothing's really going on. After hours of work the attention of the screens get lower and lower and your eyes gets more lazy. The hours continues and it's time to go home, a calm night you think and walks home. The next morning its known that there was a break-in in one of your areas which you missed and several valuable items were stolen."

- **Scenario 3**
  "You are an operator of a surveillance company which have cameras all over the city. The cameras are located in many different locations and directions. You get a call from a guard that there is a burglary going on at Nobeltorget. Since you are quite new to the position, not all the locations of the different cameras are known and you have to go through the list to see what's viewed. Because of the bad weather the cameras image is quite blurry and it takes some time to localize where the camera is positioned. A good thing that it's still daylight, otherwise it would be even worse. After a few attempts to view the correct camera you finally get it right. Luckily this time, nothing bad happened and everyone is safe, but in a different situation, things could get worse."

- **Scenario 4**
  "You work at a surveillance company which have several different kinds of cameras, because of the laws in Sweden you can't record anywhere you want. There is a privacy mask on the camera which blocks parts of the image. A person in the image starts a fight and starts running away, unfortunately in a direction where you can't look. There are thermal cameras out there but since it takes time to find them, the fighter escapes."

- **Scenario 5**
  "You are a guard at a large mall in the city and your job is to track down shoplifters. With your education in the matter you know that the thieves walk in a certain pattern and have a characteristic behavior. You are just one man and

can only be at one place at the time so when the shop assistant calls in a theft it's almost impossible to track down the thief that blends in well in the crowd. If only you had someone, or something else to track persons movement so you can be at the correct location when theft occurs."

3.3 Brainstorming and Bodystorming

The brainstorming technique was used to get new ideas in a spontaneously way. The need for brainstorming or bodystorming was to strive for and welcome unusual ideas, but also to combine and improve earlier ideas. The result of the brainstorming was written down or sketched for later analyze. When the base of the project took form a lot of time was spent on brainstorming but also the bodystorming sessions was an important tool in the project. With bodystorming scenarios and ideas could be stage-manage. Since the project is based on surveillance the need of thinking out of the box was helpful. Thought towards playing the role of a supervisor of a camera system was performed but also to be play a person that is breaking the rules, breaking into a forbidden area or just being detected performing some strange behavior.

3.4 Storyboard

Storyboard is a graphical illustration that has the looks of an comic strip. The main idea is to play out a scenario on a piece of paper. Storyboards were used to create a graphical and concise description over the ideas of the different scenarios in the project. Storyboards are simple and democratic since they are cheap to create and persons with different backgrounds can be a part of designing them. The storyboards created can be seen in Figure 3.3, Figure 3.4, Figure 3.5, Figure 3.6 and in Figure 3.7

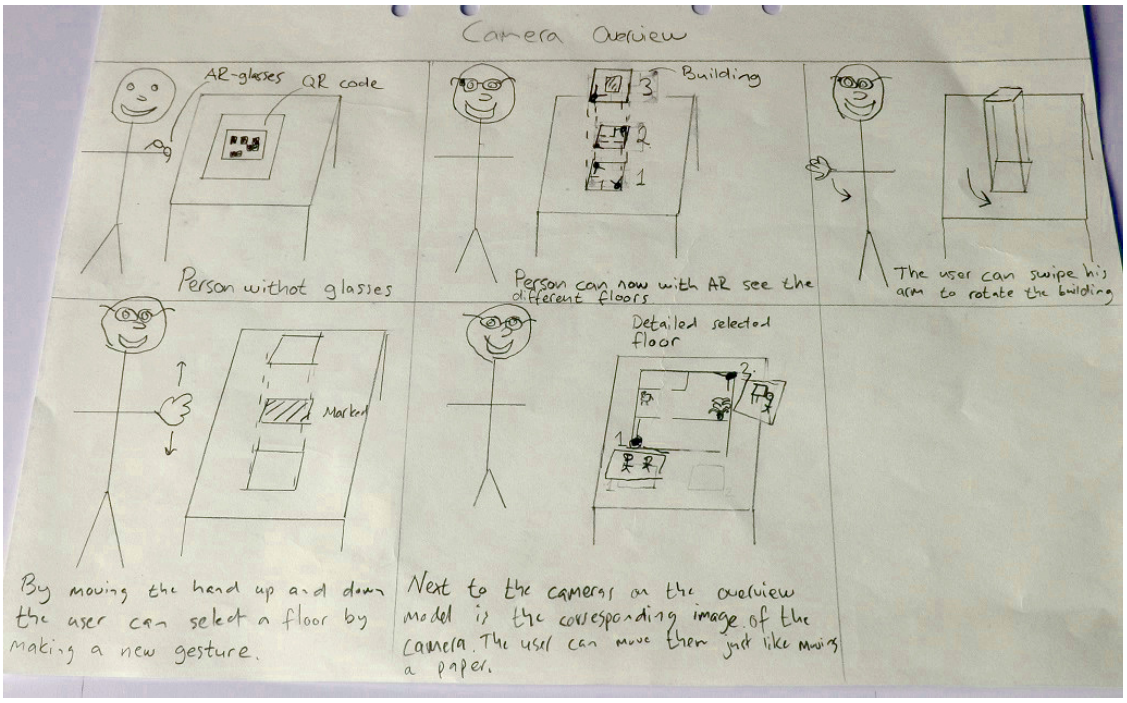

Camera Overview
AR-glasses
QR code
Person withot glasses
Building
3
2
1
Person can now with AR see the different floors
The user can swipe his arm to rotate the building
Marked
By moving the head up and down the user can select a floor by making a new gesture.
Detailed selected floor
Next to the cameras on the overview model is the corresponding image of the camera. The user can move them just like moving a paper.

FIGURE 3.3: Storyboard with an overview of a building.

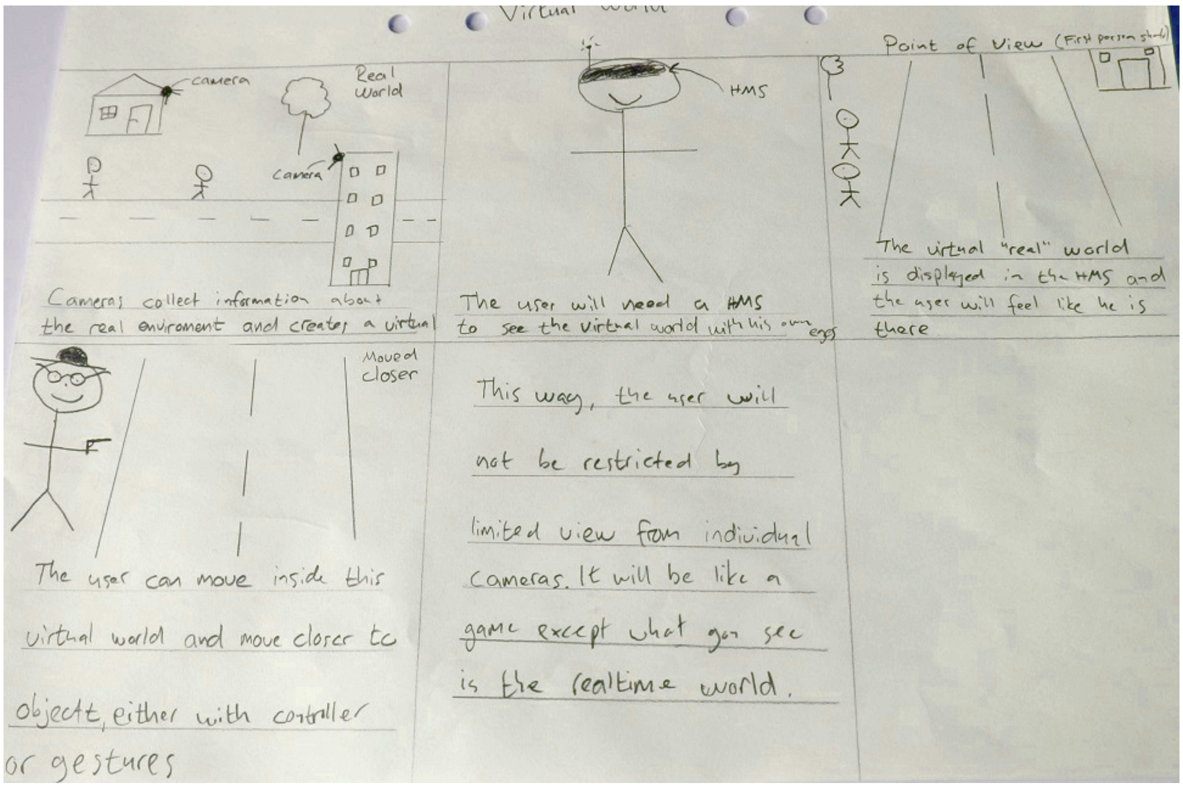

Virtual world
camera
Real World
Camera
Cameras collect information about the real environment and creates a virtual
HMS
The user will need a HMS to see the virtual world with his own eyes
Point of View (First person shot)
The virtual "real" world is displayed in the HMS and the user will feel like he is there
Moved closer
The user can move inside this virtual world and move closer to objectt, either with controller or gestures
This way, the user will not be restricted by limited view from individual cameras. It will be like a game except what you see is the realtime world.

FIGURE 3.4: Storyboard with a virtual world created from reality.

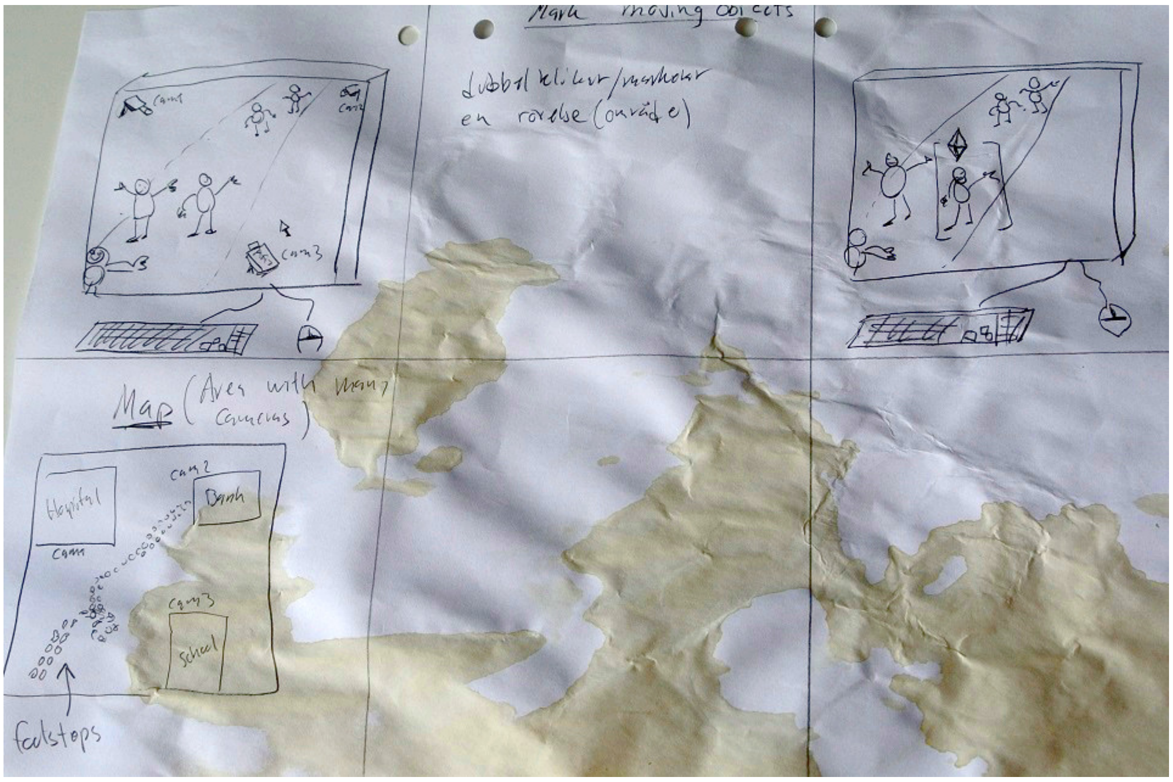


Figure 3.5: Storyboard with a moving object which is tracked. (with some improved coffee stains)

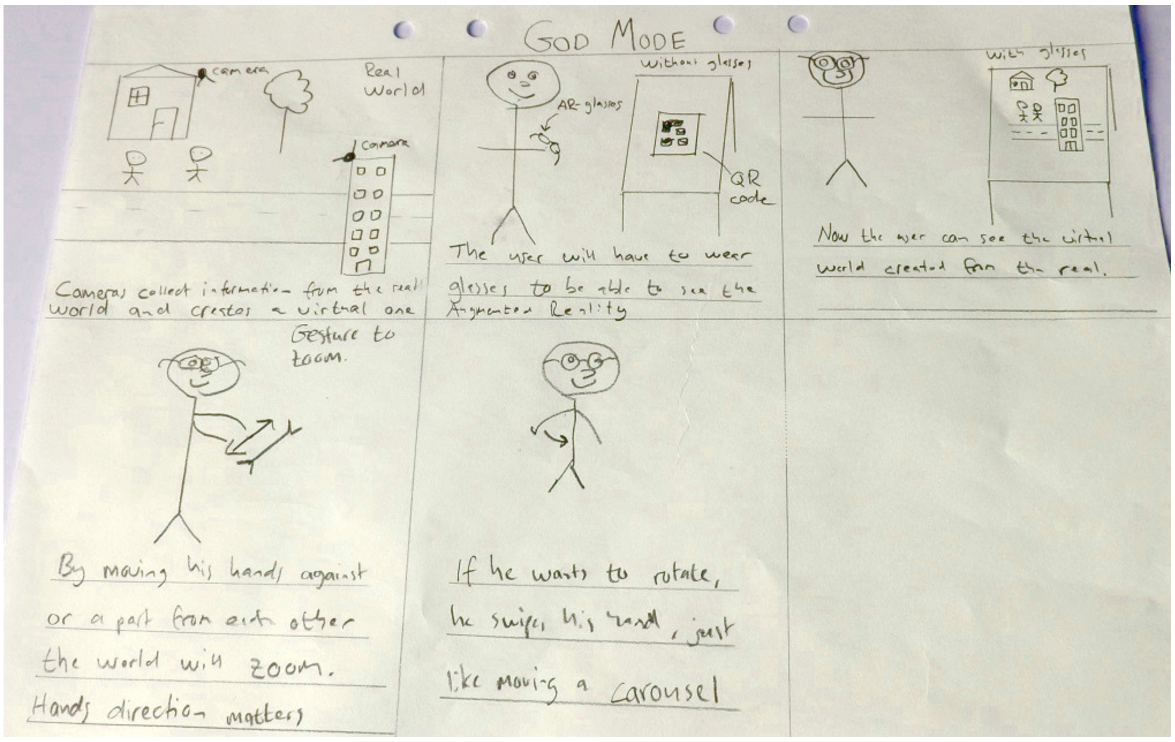

God Mode
camera
Real World
camera
Cameras collect information from the real world and creates a virtual one
without glasses
AR-glasses
QR code
The user will have to wear glasses to be able to see the Augmented Reality
with glasses
Now the user can see the virtual world created from the real.
Gesture to zoom.
By moving his hands against or a part from each other the world will zoom. Hands direction matters
If he wants to rotate, he swipes his hand, just like moving a carousel

Figure 3.6: Storyboard with an overview from above.

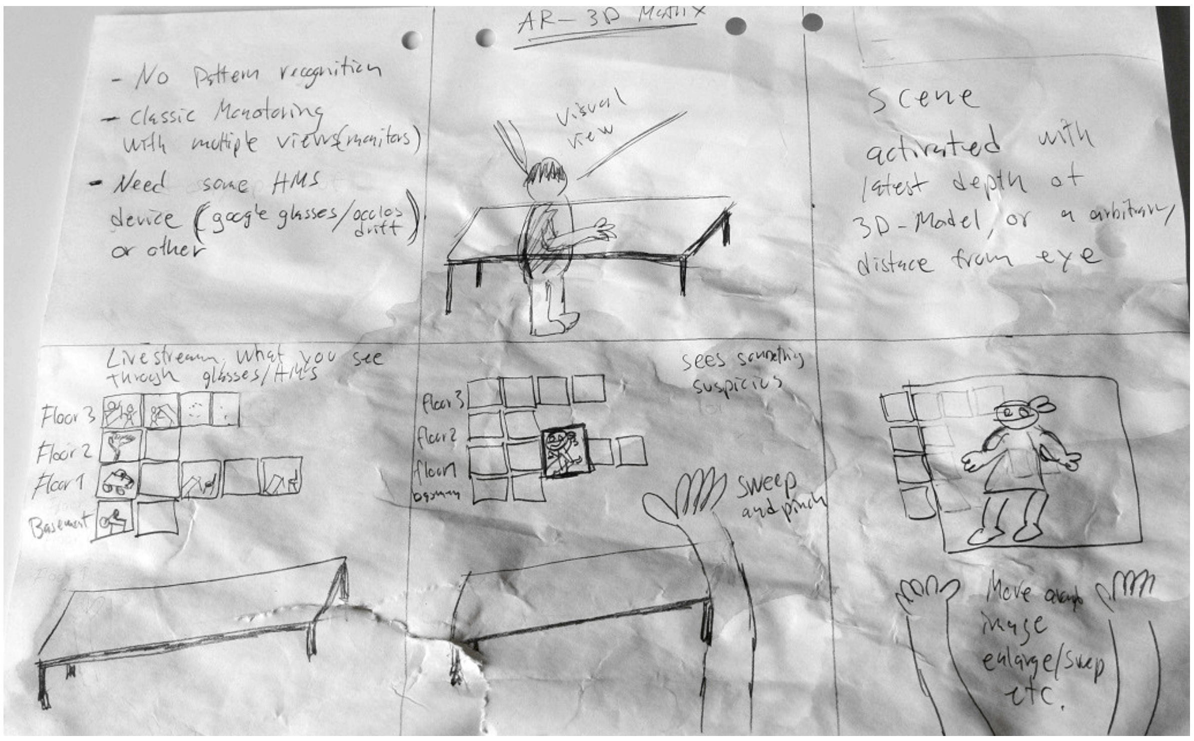

AR – 3D Matrix
- No Pattern recognition
- Classic Monotoring with multiple views (monitors)
- Need some HMS device (google glasses / occlus drift) or other
Visual view
Scene activated with latest depth of 3D-Model, or a arbitrary distace from eye
Livestream, what you see through glasses/HMS
Floor 3
Floor 2
Floor 1
Basement
sees something suspicious
Sweep and pinch
Move around image enlarge/swep etc.

Figure 3.7: Storyboard with an overview of many cameras as a matrix.

3.5 Conceptual Visualizations

Some of the ideas that was described earlier in section 3.2, section 3.3 and in section 3.4 were visualized. Visualization of conceptual ideas is a simple way to show the possibilities of ideas that are somewhat abstract. Visualization of the ideas can also lead to the inclusion or exclusion of ideas later in the implementation phase of the project. This because some ideas can seem to be hard to implement, but might come easy during later steps of the implementation phase.

Different conceptual visualizations was demonstrated and the *face tracking* is one of them, seen in Figure 3.8. When face recognition is applied, augmented face tracking method can be powerful by marking out or putting information about certain access or other crucial information. The information may be about employees and their access level.

Face Masking can be augmented and visually improved with graphics and instead of earlier when blurring of whole areas around the face was performed, the area can be covered with a graphical mask. A graphical mask that makes the perception of the image visual enhanced as illustrated in Figure 3.8.

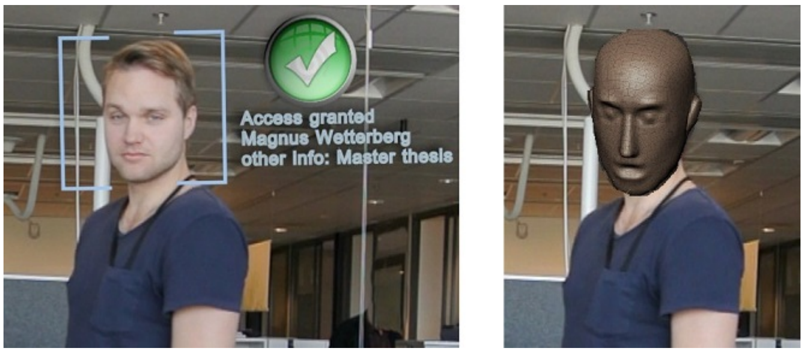


FIGURE 3.8: Face Tracking, adding text, graphical information or face masking.

Another visualization is about *Painting tracked movements*. The idea is based on following crowds of people and to mark out where there is a lot of movement. The supervisor of the camera system can easily get an overview of people's motion and movement-pattern. Information of how single persons or hoards of people are moving can easily be visualized, illustrated in Figure 3.9.

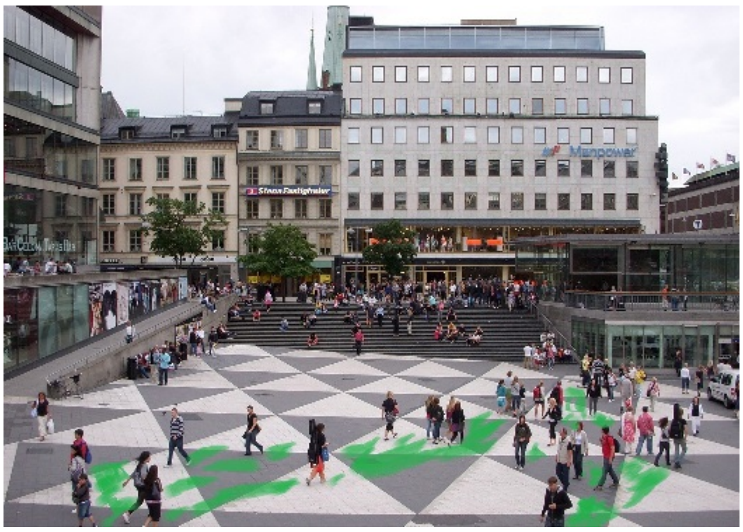


FIGURE 3.9: Movement tracked and painted.

Visualization of the *create map over tracked movements* can be of great interest for the supervisor. To start marking out one person in a camera, that movement can be followed and marked throughout a city or mall with the help of other cameras. The result and benefit of this person-movement-tracking is that the movements can be followed in real-time, but also view on a saved map for later review. The idea of marking out a person and to make use of multiple cameras that can help each other out to map a moving person is illustrated in Figure 3.10.

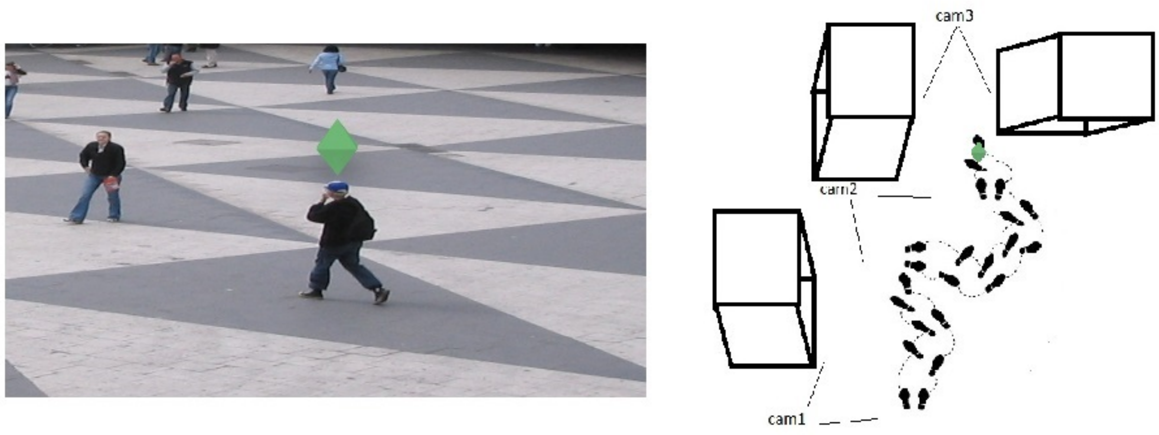


FIGURE 3.10: Person to be tracked selected and a map of a tracked person.

Create a 3D world from real world images was visualized in Figure 3.11 to show the thought of identifying parts of the real world and replace them with create 3D object. This could be a benefit when permission to monitor is needed. It is also useful to mark out objects of interest in a graphical way.

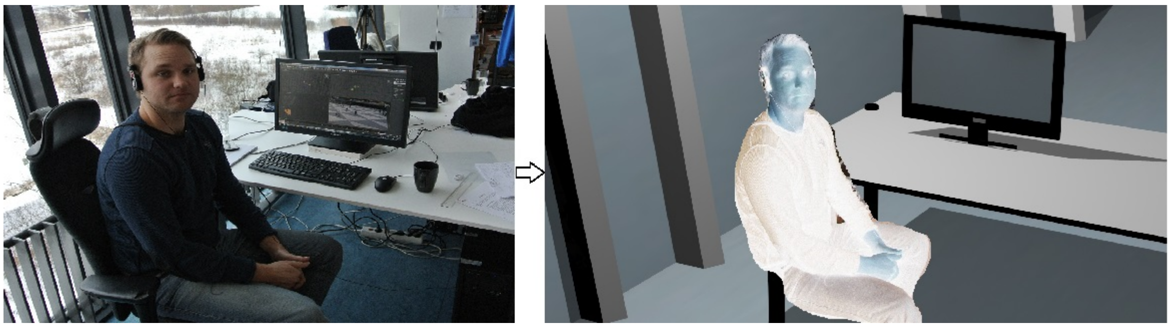


FIGURE 3.11: Real world to the left, 3D world with movements from the real world to the right.

Another visualization is the *overview with HMD* where peculiar ideas came up. Since Google glasses is on the wall many ideas lead towards the use of letting our own eyes work as cameras with additional information added in form of real time videos. During this part of the conceptual visualization many ideas was to be thought of when implementing later on, ideas of getting an overview of many cameras in 3D and collect multiple camera streams in one image, see Figure 3.12 and Figure 3.13.

One idea for HMDs usage is for a guard guarding in a mall. The guard is equipped with a see-through-HMD and with the HMD-device the guard gets a ”superman-vision”. The guard is able to receive any image from any cameras in the mall. By looking in any direction towards a camera on the same floor or on another floor the view of that camera

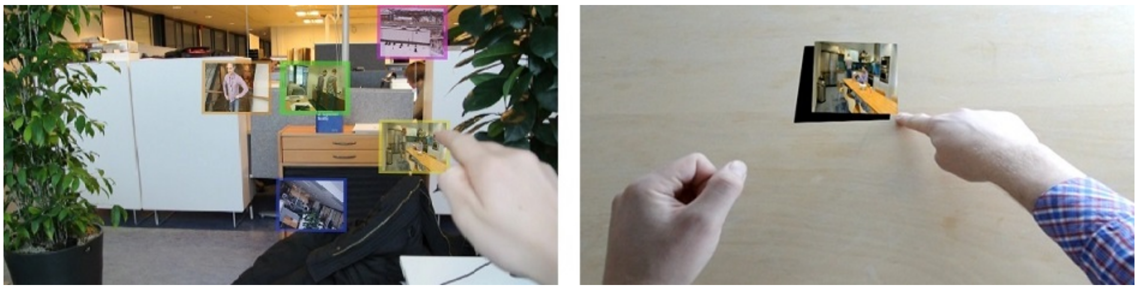


FIGURE 3.12: Moving around a real time video stream.

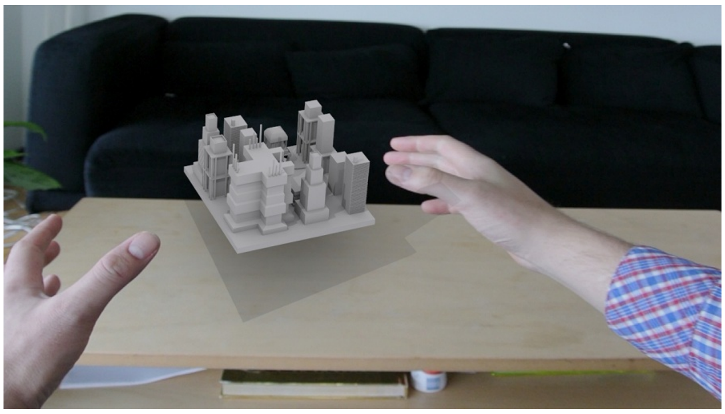


FIGURE 3.13: Swipe to move around buildings.

can be shown directly on the HMDs screen. In this way the guard could be standing on one floor down in the mall and to be able recieve an image from any floor by lifting his head towards a direction.

3.6 Choice of product - Axis Maps

As a result of the previous phases, one single product was defined which combines the best parts of each idea. This will be the application to work with and implement from now on.

As many previous ideas indicated, it is necessary to have an overview of the system. Instead of using a folder or a list to show the cameras, visualization is made by using Google Maps. The cameras will have a GPS-position and be placed on the map by their location and the user can interact with the objects directly from the map. Since the camera interface is now graphical, it is now important to also know which direction the cameras are pointing at. With the new overview, the operator can easily find the specific camera by knowing its location. To make the process easier, it is also possible to filter by the cameras type. A visualization of the camera list has many other advantages. Since the cameras are now displayed on the map, the operator won't need to memorize which camera that is on the specific location, the map holds that logic. The overview will be the place to configure cameras. Cameras that will be used later on by the AR part of the application.

To make the orientation and navigation easier when viewing the camera, the additional information of other cameras position and direction are now used. By using AR, it is possible to add information of the other cameras in the image, on their correct position. In practice, if a camera is looking in the direction of another camera, it will be displayed with a textbox exactly where the actual camera is. This will give the operator higher orientation but not enough. To increase it even more, information of buildings, streets and other structures will also be placed in the image. These added information is called *Point of Interests* (POI) and are specific for each camera. The POIs will be added to a camera in the Map View, the Map View that holds the overview of all the cameras in the system.

The POI boxes with information which are now placed in the image will also be interactive. This makes it possible to click on them and by the type of the POI, get the desired effect. If a camera is clicked, it is desired to enter the camera, if a building or road is clicked, it might be more important to get other cameras viewing that specific POI. It might be cases where the user rather wants to view the cameras inside the building by clicking on the box, therefore a camera would also be placed to be able to navigate there. Using the interactive interface will simplify the navigation of the system, since the operators do not need to go to the overview of the cameras if it's not necessary.

To summarize the choice of product, *Axis Maps* is new way to get a full overview of a camera system along with computer graphics to enhance the image of the cameras.

Chapter 4

Design

4.1 Requirements and features

To define the product and its content a list of requirements and features was created. With limited time in the project, the requirements needed to be divided into sections dependent on the value for the product. The sections are grouped and prioritized in the following order: *Necessary*, *Desirable*, and *Unnecessary.* The list which also displays which has been implemented can be found in Appendix A.

4.2 Lo-Fi prototype and exploratory testing

A Lo-Fi prototype was built to test the design on users. The prototype also gave us the possibility to get an early input and to get a grasp on how the final design would look like. A Lo-Fi prototype has many advantages. A Lo-Fi prototype is easy to create, cheap and democratic. Democratic in the way so that everyone can participate. The prototype can be built in several different ways including using software programs such as Balsamic Mockup [18] or Microsoft Sketch Flow[19]. Even though these products are good and well-designed scissors and papers were chosen. The Lo-Fi prototype resulted in two different views, the *Map View* displayed in figure 4.1 and the *Camera View* displayed in figure 4.2.

4.2.1 Map View

The *Map View* will provide an overview for all the cameras on a map. The cameras are displayed over their position with different symbols which makes it easy to separate the different types of cameras. In some cases the map will display multiples cameras and therefore a checkbox selection has been added. In this way different cameras of no interest can be hidden.

If the user touches a marker a window will show up right next to it. It is from here the user can edit or delete the cameras and also add to POI specifically to the chosen camera. To create a new POI the user only has to press on the map where it should be located and then add necessary information.

The Map View is where all the configurations is made and this view will be the base to the AR part of the application. In order to open the camera view the user need to press the "View" button in the window. The result of the Lo-Fi prototype of the Map View is shown in Figure 4.1 below.

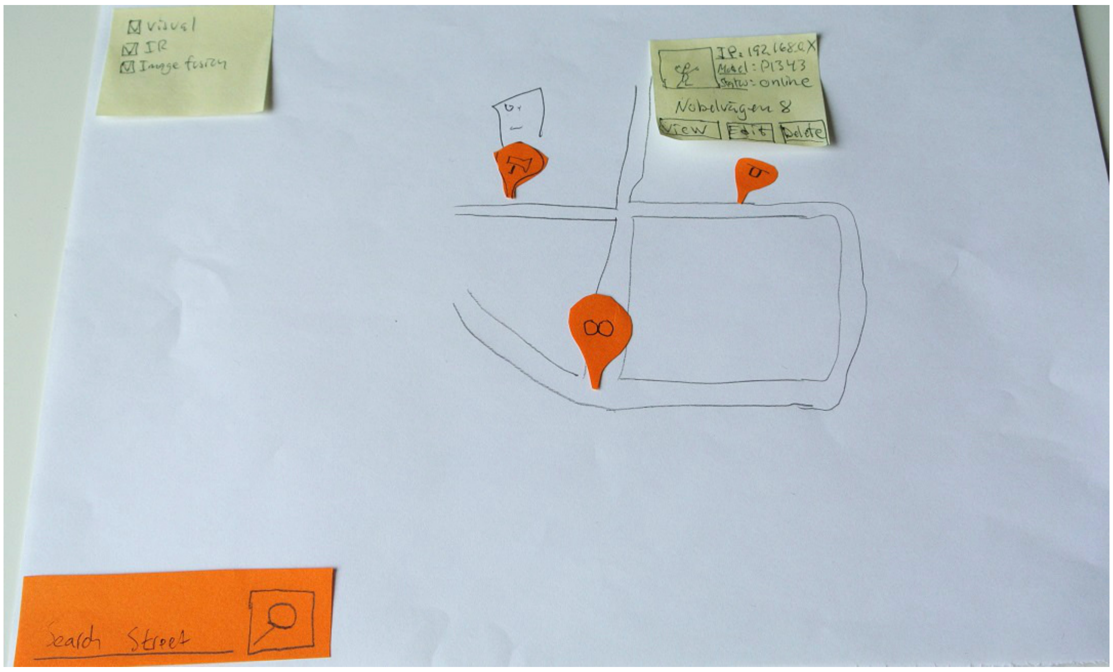


FIGURE 4.1: Lo-Fi prototype over the Map View.

4.2.2 Camera View

The *Camera View* is the part of the application which will add AR to the camera image. Every camera has their own POI and these will be displayed as billboards in the image based on their location. Therefore no image analysis is needed to track any marker or pattern and this results in a marker-less AR application. From the location of the camera and the POIs and the direction of the camera in mind the position in the image can be calculated. When the camera rotates the billboard will follow the object which the POI is positioned to.

Thanks to the display of the smartphone, gestures can be added to perform pan, tilt or to zoom on the camera. With a swipe the user rotates the camera in the same direction and a pinch will zoom the camera.

If a billboard is clicked other cameras viewing the same billboard will be displayed. This simplifies navigate between several cameras that are watching the same area. The result of the Lo-Fi prototype of the Camera View is shown in Figure 4.2 below.

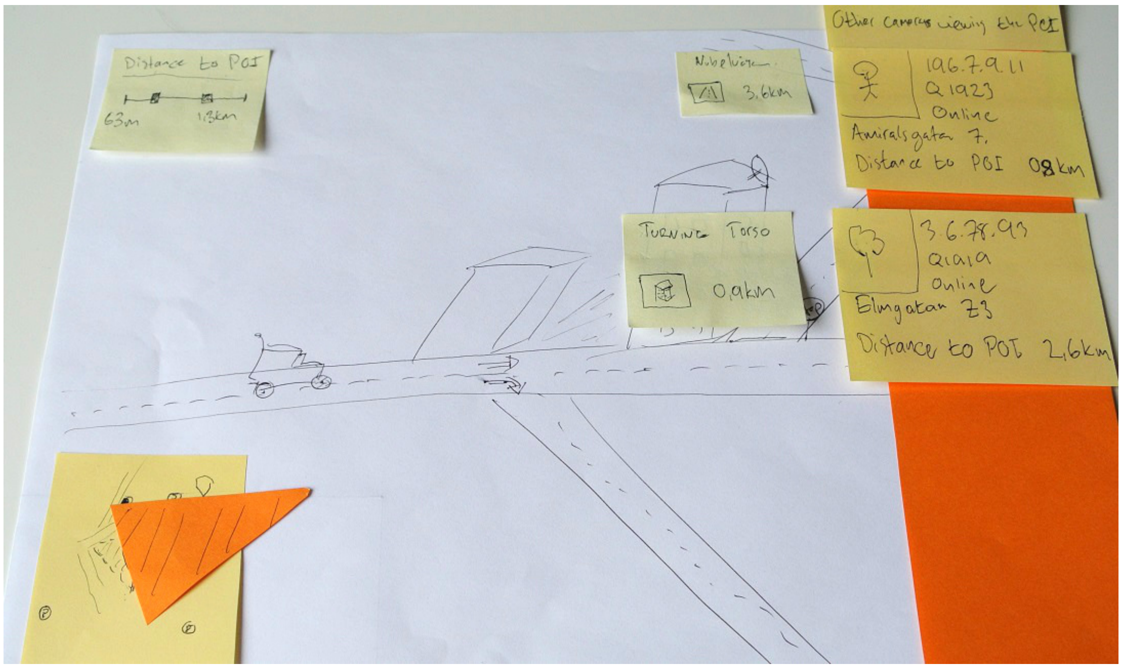


FIGURE 4.2: Lo-Fi prototype over the Camera View.

4.2.3 Exploratory testing of Lo-Fi prototype

To get an early input from the users a test was created for the Lo-Fi prototype. It was tested on five persons not familiar with the purpose of the application or its features. The method that was applied let the tester think out loud of every thought during the interaction with the application. There were no given goals to achieve for the user and the only directions were to use the application. The result of the exploratory tests lead to some good parts and some not so good parts. The result is listed below.

Good

- Configuration of cameras. Edit, delete and add cameras was clear.
- View a camera.
- What the billboards was displaying and orientation of where the camera was located.
- Sort through different camera models.
- Gestures to pan, tilt and zoom the camera.

Not so good

- The users wanted to display POI in the Map View instead of cameras and choose camera dependent on the POI and not the other way around.
- What happened if they clicked on the POI in the camera view. Some thought the camera would focus the POI and no display different cameras as intended.

With the user input the application design was changed to ensure their needs were fulfilled.

4.3 Hi-Fi prototype

The Hi-Fi prototype is the final result of the design and implementation of the application. During the implementation and design of the Hi-Fi prototype thoughts towards choice of design such as affordance, mapping, visibility, color and size was made. The Hi-Fi prototype contains of *Map View* and the *Camera View* which is the two main interfaces of the application.

4.3.1 Map View

Navigation in the Map View is analogously to the usual way of moving around on a screen and since Google Map uses this kind of navigation it works according to Google maps standard implementation. Touching the screen and dragging in any direction will move around the view, as showed in figure 4.3. Pinching on the screen is another analogously way to interact in the Map View. When pinching the image is being zoomed in or zoomed out, as can be seen in figure 4.4. The overall design is made to avoid unnecessary graphics or unnecessary buttons. The design strives to be simple as a simple click or a click that is being pressed down for some time to add object or to open up informative windows.

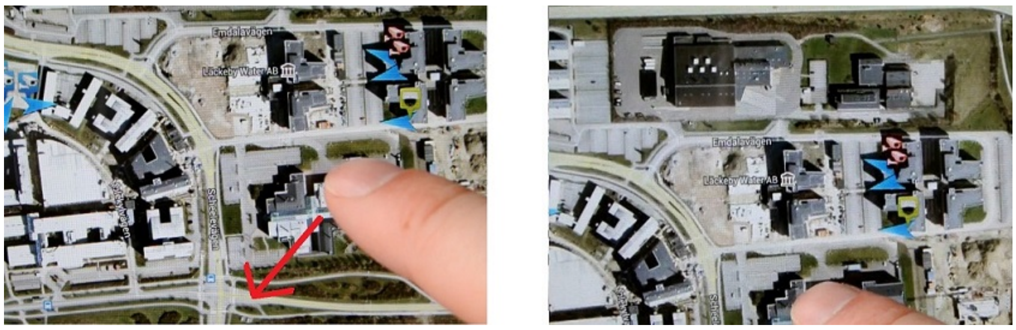


FIGURE 4.3: 1. Touch and hold the screen while swiping. 2. Lifting finger from screen when done.

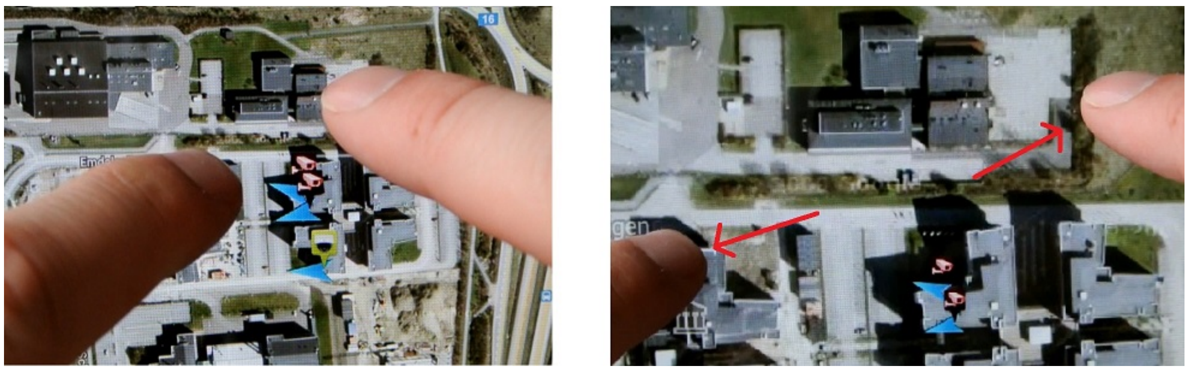
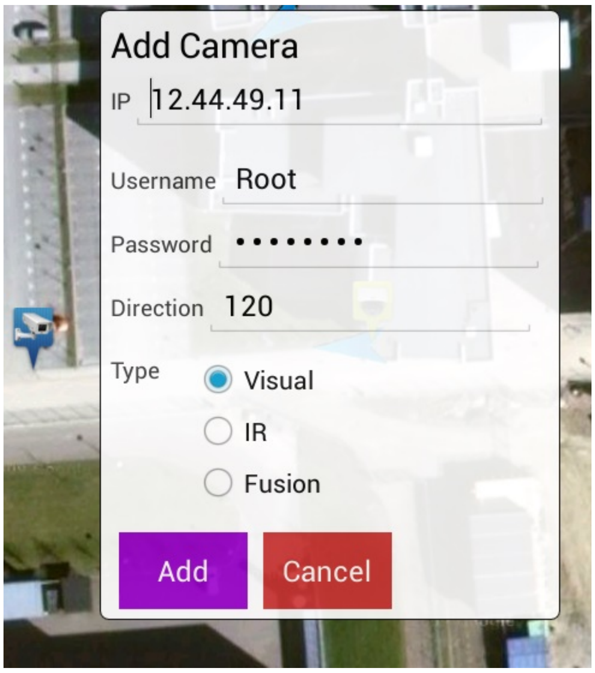


FIGURE 4.4: 1. Using two fingers to press on screen. 2 Dragging fingers in separate directions to zoom.

Adding a camera is done by clicking on a arbitrary position on the screen and then keep holding down for some seconds. A window of adding camera information turns up, shown in figure 4.5.


Add Camera

IP 12.44.49.11

Username Root

Password ••••••••

Direction 120

Type [✓ Visual] IR Fusion

FIGURE 4.5: Camera information.

The icons of the cameras where designed to have colors that somehow represent the camera itself. The blue color was chosen to represent a static RGB camera and the red color to represent the infrared camera. The green was picked because of the PTZ's agility. The color of the Image fusion camera was a mixture of blue and red due to the mix of infrared and RGB-camera. The different cameras is shown in figure 4.6.

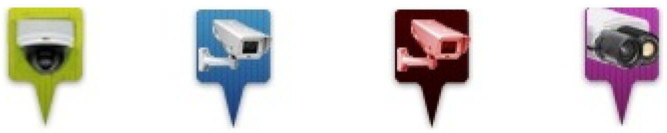


FIGURE 4.6: Camera information.

When adding a camera the direction of the camera is added and marked as a blue arrow (Figure 4.7). The direction-arrow was designed to point out the live-view direction of the camera.

Figure 4.7: Arrow of camera direction.

In the design of the configuration view of a camera, three features in the shape of buttons where added. The first button is to switch to the view of live camera image, second button to edit the existing information of the camera and the last button to add POIs, shown in figure 4.8.

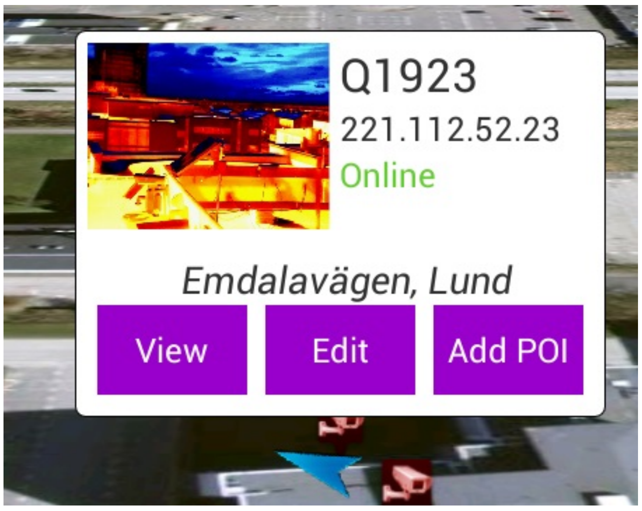


Figure 4.8: Camera window in Map View.

When clicking on an arbitrary position in the world the three closes cameras update their direction and turns towards the clicked position. The popup window was designed using colors that separate the different cameras. The colors are connected with the colored lines between the cameras and the position, this to clarify which camera view that belongs to its live view, shown in figure 4.9.

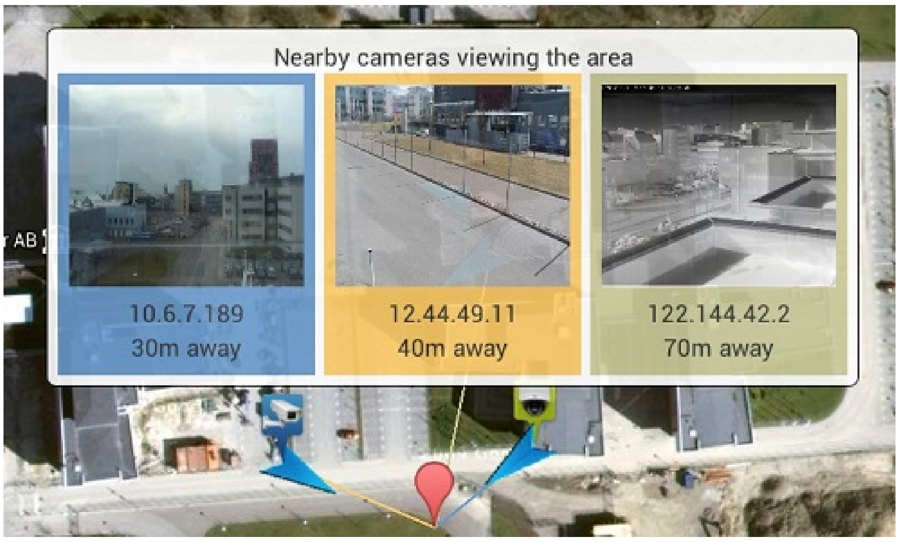


FIGURE 4.9: Popup window that present the three closest cameras to a point.

4.3.2 Camera View

The Camera View is the live view of a camera. In this view other cameras, a mini map of the cameras direction and POIs is visible, as can be seen in Figure 4.10. In the Camera View motion detection can be performed and the tracking of moving object is being marked with an icon.

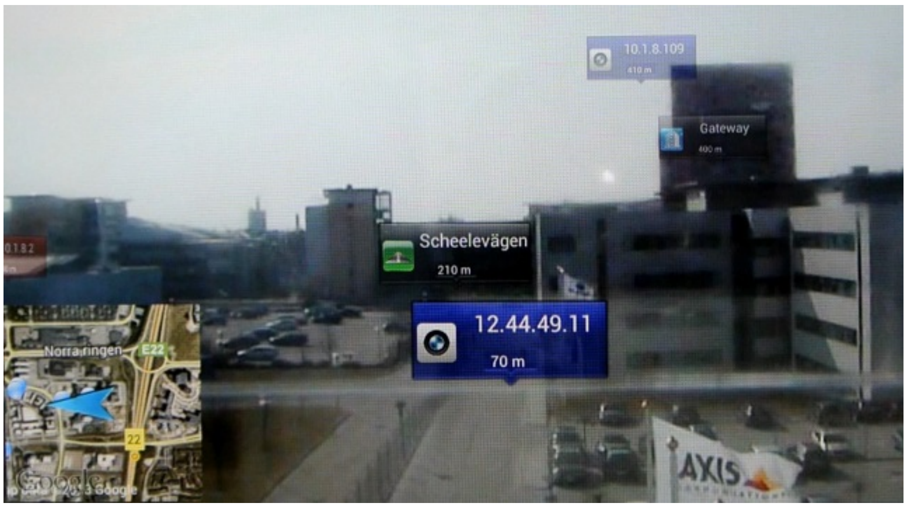


FIGURE 4.10: Camera view.

The Navigation in the Camera view is designed with natural movements in mind. The navigation in the Camera View only relates to the PTZ-cameras due to the PTZ-cameras ability to change direction. When swiping to the left or right the built in camera engine is rotating the camera and the image is being updated to the cameras position.

Cameras and POIs in the camera view present themselves as billboards, shown in Figure 4.11, and in Figure 4.10 demonstrated in a real scene. A billboard represents a POI that contains information of which building or street it belongs to. If the billboard represents a camera it contains camera information and has the same color as the camera icon in the Map View, to keep consistency.

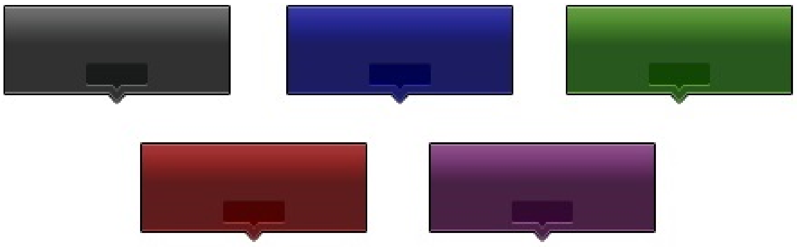


FIGURE 4.11: Couple of billboards.

When motion detection finds an object it marks the object with an red icon and when handing over information about tracking to another camera the green icon is used, as can be seen in figure 4.12. To catch the viewer's attention, the icon of the motion detection changes in size and changes in color when one camera hands over the tracking data to another camera. Figure 4.13 shows how the icons is placed on a object.

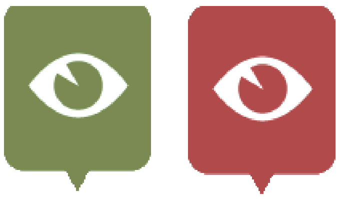


FIGURE 4.12: Motion detection icons. Green marks when handing over to another camera. Red icon marks the motion.

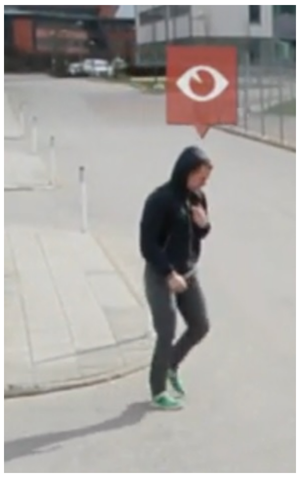


FIGURE 4.13: Marked and tracked motion with a red icon.

The mini map shows the camera views direction in the world. It also presents the cameras POIs marked with blue dots, shown in figure 4.14. The mini map is clickable and returns to the Map View.

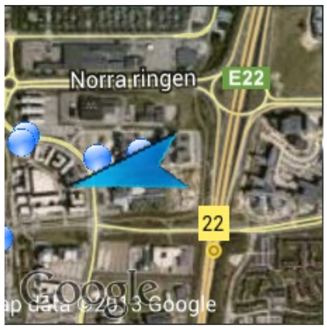


FIGURE 4.14: Mini map in the Camera View.

4.3.3 User feedback

An exploratory user test of the Hi-Fi prototype was conducted among employees from different teams on Axis Communications (together around 20 persons). The feedback was positive and the project was given a lot of appreciation. More indication pointed towards the "reverse POI" which been thought of earlier, but was to be implemented after the meeting. Some new inputs were given and the idea and ability to insert own maps came up. Adding own maps made way to be able to work with internal maps of buildings. The possibility of working with own maps and adding different floors on a building was the final conclusion of the meeting.

Chapter 5

Implementation

5.1 Review of AR-Tools

There are several tools/SDKs to achieve AR and many has their strength but also weaknesses. This section will compare some of them and validate which is most proper for the purpose of the AR implementation. The review will also look into the possibilities to integrate Google Maps API[1] or any similar API with the tools and SDKs. The most important features which will be evaluated are:

- Which platform are the tool available for?
- Is it free and does it require any additional application such as Unity and Silverlight?
- Can it be connected with an Axis camera?
- Can it be integrated with Google Maps or similar map-API?
- Which tracking methods are there?
- Are there well documented instructions and tutorials?

[1]"API is a specification of how some software components should interact with each other.", Reference: `wikipedia.org/wiki/Application_programming_interface`

These are the following tools and application which were compared and analyzed.

5.1.1 DroidAR

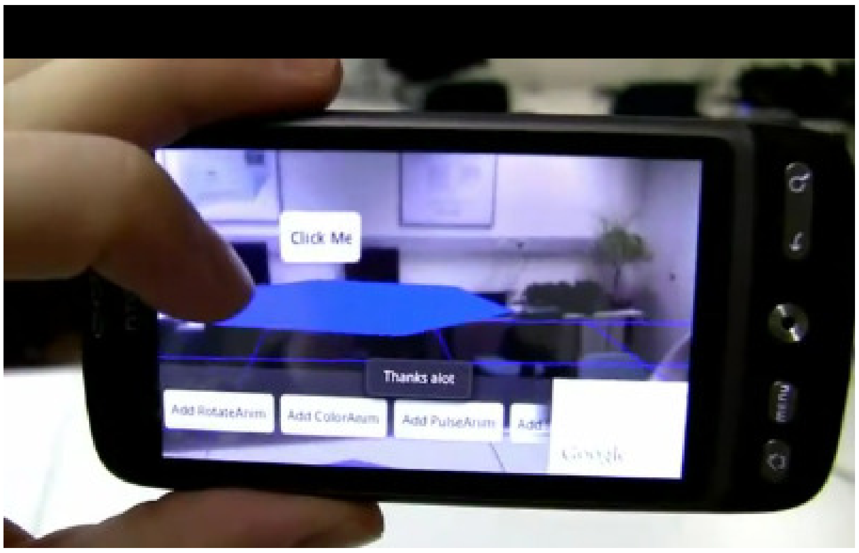


FIGURE 5.1: DroidAR with some graphics and an interactive button.

DroidAR [20] was created as a bachelor thesis by Simon Heinen in 2010. Since then a community has been created around it as an open source project. There are a few developers who still work with the framework and use it in different application but is not one of the most popular. It is built under GNU GPL v3[21] which integrates the camera image with the 3D world and creates the AR.

Which platform are the tool available for?
Android

Is it free and does it require any additional application such as Unity and Silverlight?
DroidAR is open source, free and does not require any additional applications.

Can it be connected with an Axis camera?
Yes, with the open source code can the camera input be modified but it requires some effort and knowledge. Since it is not only the image that will need to be replaced but also the sensors this can be quite hard and requires many changes to the code.

Can it be integrated with Google Maps or similar map-API?

Yes, the framework supports POI with GPS coordinates but don't have an inbuilt billboard to use. This would be needed to be implemented.

Which tracking methods are there?

The framework only supports markerless tracking and uses the camera sensors to create the AR.

Are there well documented instructions and tutorials?

The code is well documented and quite easy to learn and understand. However, since the framework is not well known by others there are not any tutorials or documentation except from the one that the owner created which is not much. There is no forum to ask questions so everything must be handled by ourself.

5.1.2 Metaio

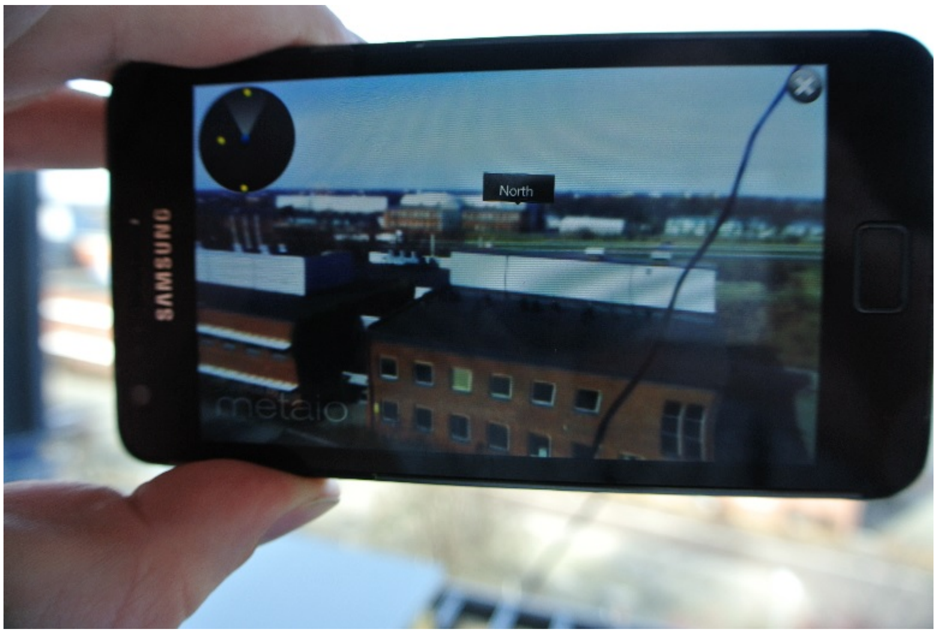


FIGURE 5.2: Metaio with an billboard set to north.

Metaio [22] is one of the most popular and used AR SDK. It was founded in 2003 by a large company and has won prizes such CEBIT Innovation Award (2004) and the tracking contest at ISMAR [23]. Their goal is to make money and therefore the SDK is not open source.

Which platform are the tool available for?
The SDK is available for Android, iOS and Windows.

Is it free and does it require any additional application such as Unity and Silverlight?
The SDK is not free; however it can be used without any cost with the disadvantage of a watermark and load screen. It has been integrated with Unity for both Android and iOS but requires Unity Pro. The Windows SDK requires Windows Visual Studio which is not free.

Can it be connected with an Axis camera?
No, since the SDK is closed the camera and sensor input are unchangeable and cannot be modified.

Can it be integrated with Google Maps or similar map API?
There seems to be no easy way of adding POIs with GPS coordinates to the location based application. The POI has to be predefined in a separate tracking file which is later loaded to the application. If this can be accomplished, the billboard is already implemented and can be used.

Which tracking methods are there?
Both marker and markerless tracking with the exception that the markerless tracking has been removed from the Windows SDK.

Are there well documented instructions and tutorials?
There is some example code on how to use the SDK but the documentation is poorly written and hard to find. Some tutorials have been made but don't cover all the needs. There is a forum to be used which is frequently watched from other users for questions.

5.1.3 Mixare

Mixare [24] is an free open source AR engine and works as a completely autonomous application for Android and iPhone. Mixare is also available for the development of own implementations.

Which platform are the tool available for?
Mixare is available as an app for Android and iPhone

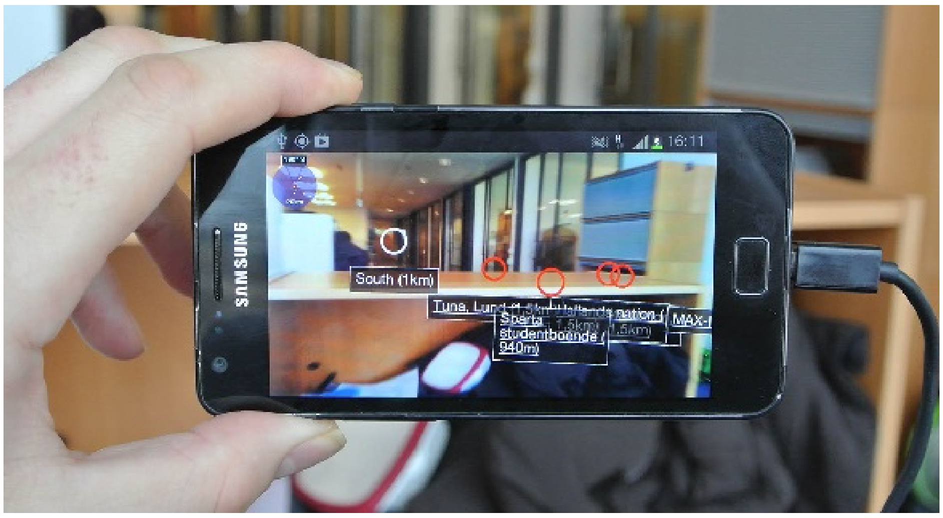


FIGURE 5.3: Mixare looking to the south and southwest.

Is it free and does it require any additional application such as Unity and Silverlight?

The Mixare is free to use and free of future development because Mixare is open source.

Can it be connected with an Axis camera?

Not in a simple way. Mixare uses the Android or iPhone built in camera and to be able to use a IP-camera such as one of Axis calls for a greater change in the code.

Can it be integrated with Google Maps or similar map API?

In its initial setup Mixare uses public POI information from Wikipedia and twitter. Mixare works with Google Maps but it is not easy to add your own Google Maps information, but fairly easy to add own POIs.

Which tracking methods are there?

Markerless tracking is available.

Are there well documented instructions and tutorials?

There is no updated tutorial. New code has been added and some of the tutorials are written for an older version of the application that does not look anything like the latest one. The documentation is not very well and it is hard to find out where and what you can change in the code for simple changes like adding POI.

5.1.4 Choice of Tool

Since none of the AR SDKs or open source AR-engines was easy to use or rebuild for the project, the decision was made to implement the AR-application on our own. It was important to use Axis own cameras, something that the SDKs could not provide. The decision also favors the coding since no time will be spend learning the existing structure in the SDKs or in the open source engines. The own implementation will also give full freedom in design choices which is an important part of the application. The platform chosen was Android which has great support of Google Maps. The own implementation of the AR-engine will be able to receive a stream of images from an Axis camera. Android also has plenty of documentations and tutorials that are free to use.

By using the platform Android, the interaction method used will be touch, which gives many advantages. It is a platform which is suitable for Axis systems and one that does not seem too futuristic. This is also why smartphone/tablet was chosen since HMDs is yet too unexplored and not really used in surveillance, yet. It is important that the system is easy to configure and can be used in a greater system, therefore a location-based technique is used which will rely on the sensors from the camera and will also show the advantages of using Google Maps in a system.

Many of the previous ideas were aimed in a different direction which is more intended in the future. But it is important that the implementation can be made of and used today.

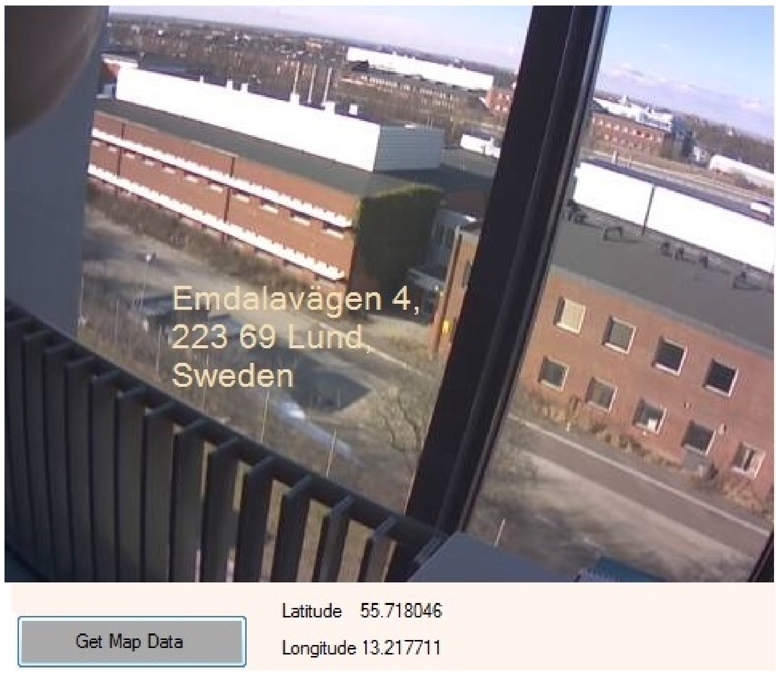
5.2 Platform

Since none of the AR SDKs or open source AR-engines was easy to use or rebuild, as mentioned in 5.1.4, the project and that the SDKs or AR-engines could not provide support in a simple way for using a input-stream from a IP-camera the decision was made to implement the AR-application instead of relying on some existing AR solution.

We started out by setting up a video stream from a IP-camera in C#(programming language) and tried to get graphical functions with different OpenGL wrappers. The choice of using C# was mainly because of the Axis Media controller that has great support for playing video in a C# form. The issue with C# was that there exist no true OpenGL support and it has no easy way of handling graphics by default.

One attempt to get started with the implementation was to create simple text in the image as can be seen in 5.4. The text was dynamic and could be moved around in real-time. The text implementation was done so that more effort could be put in the Google Maps environment and work with logic of the Google Maps such as POI, GPS positions of the camera, virtual compass on the camera and with the cameras point of view. The texts combined with Google Maps were to form the location based application that we wanted to implement. Since there is no support for C# by the Google Maps API, a wrapper was used called Gmap[25]. The wrapper was not well documented and therefore not very usable and there were difficulties to understand and use the wrapper. The effect of getting the text to be represented as a 3D object was difficult and to create dynamic 3D-text also had its difficulties due to the need of integration of a text library, to be able to create 3D text.

The Google Maps API has great support for Android [26] and the project got a new turn heading for an implementation in Android. The use of the built in camera in Android devices was helpful when testing different SDKs. The SDKs uses the build in camera and the idea was to replace the image stream with an image stream of the IP-cameras. Some SDKs were tested but none of them was really easy to work with so the decision to implement everything from scratch with the visually backbone of Google Maps seemed as the naturally way of the project.



Latitude 55.718046
Longitude 13.217711

FIGURE 5.4: Simple text printed in C#.

5.3 Google Maps View

The Map View is the part of the application where configuration of cameras takes place. It is possible to add, edit and remove a camera and also add new POIs to the specific camera. The view uses the Google Maps API which is integrated with an Axis camera and the own implemented AR-engine.

5.3.1 Present Nearby Cameras

In the Lo-Fi prototype testing, many users pointed out that it would be cases where it would be more feasible to choose camera dependent on the location instead of the other way around. In practice, a user would only need to press on the map where they would like to monitor and then be presented with available cameras to choose from. In addition, to be able to support PTZ cameras, the closest cameras would also turn in the correct location so that the targeted area is viewed.

If the user touches the map, a new box will be visible next to the pressed area and the three closest cameras will be presenting their image and distance from the area. To allow faster orientation of which cameras that are presented, a line is drawn from the camera to the position with the corresponding color to the box. The cameras are also sorted by the distance to the camera. The result is visible in Figure 5.5 where the user can choose from three nearby cameras.

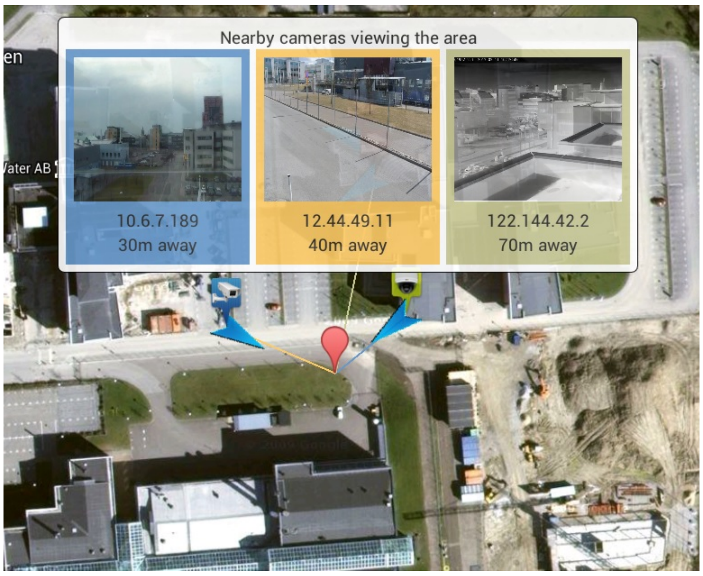


FIGURE 5.5: The three closest cameras viewing the desired area.

5.3.2 Buildings

As a follow-up on the second evaluation of the product, another dimension was needed for the map. Many customers to Axis Communications use their system in a building with several levels. To be able to get a better overview and to allow cameras on different floors something needed to be added. The solution would be an extra overlay which is the planning of the floor instead of just displaying the regular map. With the latest Google Maps API such overlay would be possible and to configure the system just requires the images of the floors, dimensions in meters and the GPS-position of where the building is located. The rest is up to Google Maps to calculate. As an addition of how to choose the floor, a new view was added to the left which includes a radio group to choose from. The result of the additional dimension is viewed in Figure 5.6.

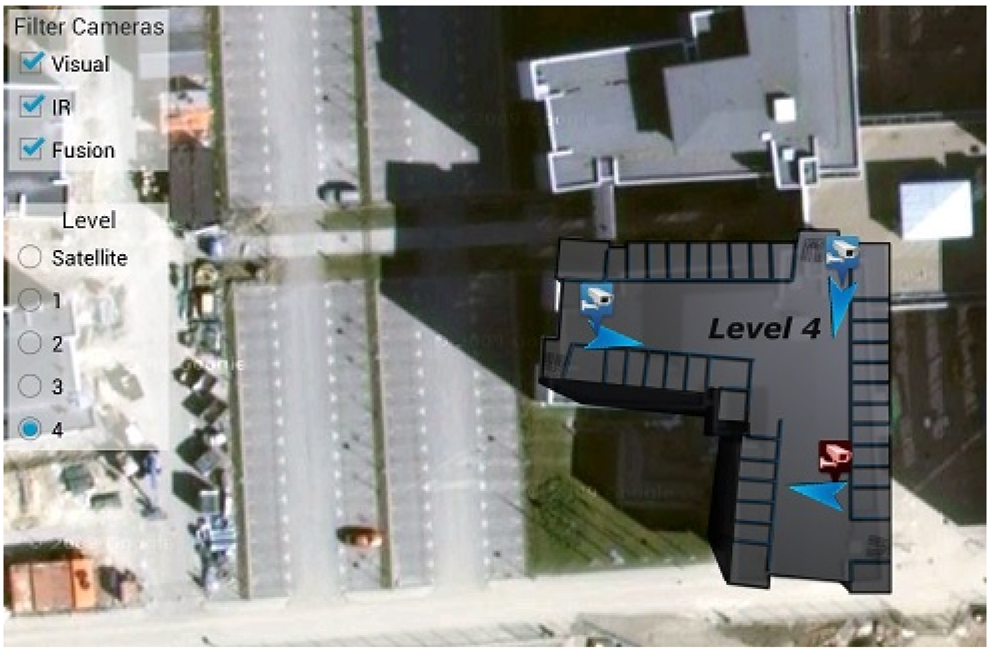


FIGURE 5.6: Google maps with new dimension of buildings.

5.4 Camera View

5.4.1 Augmented Reality

As mentioned earlier, the *Camera View* is the part of the application where AR is implemented and displayed. No separate AR SDK or API was used to be able to use Axis Communications own IP-cameras and to show that it's possible to implement in a later stage directly on the camera without any additional help.

The method used is a *markerless* tracking which depends on the cameras sensors and orientation. The GPS positioning is handled by the *Map View* where all the cameras and their POIs gets their GPS-positions. Because of the fact that the cameras are stationary and will not change location this parameter can be entered manually and no GPS-sensor is needed on the camera. As the camera is being installed, the direction of the camera is also needed. Since Axis own cameras do not have a built in compass, the direction of the cameras was also entered manually with a separate compass. Some cameras are PTZ and will change the direction as the user interacts with it, so the application needs to collect the relative direction of the connected cameras. This is done using Axis own http-based API *VAPIX* [27] which provides functionality for requesting images, controlling network camera functions and setting or retrieving internal parameter values. This way the direction of the camera and the application are synchronized. VAPIX is also used for retrieving the images to the application.

With all the necessary information retrieved the next part is to place the POI on the right position on the image. The parameters needed are the GPS-position of the camera, GPS-position of the POI, the direction of the camera and the FOV. The FOV is needed to know how wide the scene is in the camera image.

The first step is to calculate the direction from the camera to the POI. Since the earth is shaped as a globe the normal coordinate system is useless. The problem with the normal coordinate system appears at greater distances where the error will result in that the POI is placed markable off the target. Instead the Equation 5.1 is used. This equation calculates the bearing according to the globe.

$$\theta = atan2(\sin(\Delta\lambda) * \cos(\omega_2), \cos(\omega_1) * \sin(\omega_2) - \sin(\omega_1) * \cos(\omega_2) * \cos(\Delta\lambda)) \quad (5.1)$$

Where *Omega* is the latitude, *Lambda* is the longitude and *Theta* is the final bearing between the two GPS-positions.

The next step is to check if the bearing between the two GPS-coordinates is within the scene that the camera is viewing. If the bearing is within the interval of camera direction and its FOV the POI should be displayed on the image. This is illustrated in Figure 5.7 which is an illustration seen from above.

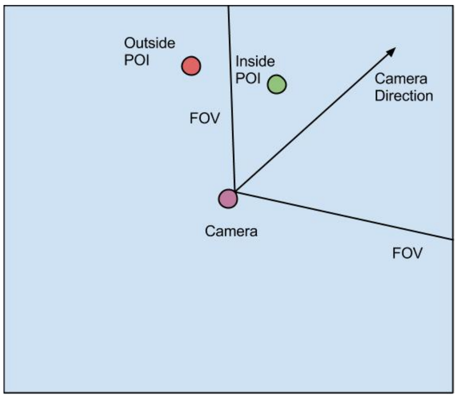


Figure 5.7: A camera with its direction and FOV which determines how much of the scene is viewed and which POI should be displayed.

To calculate the x-position of where the POI should be placed in the image the dividend between the camera direction and the FOV is multiplied, or mapped by the images width. In this way the POI will get the correct screen position in the x-axis. The mapping from the direction to the image position is viewed in the Figure 5.8.

Because of the variety of the height in different scenes its quite hard to calculate the correct y-position. In some cases the POI should land on the ground and in some cases on top of a building. Therefore the y-position is manually added by the user and can easily be modified by long-pressing the POI on the screen.

5.4.1.1 Depth Cues

AR is all about making the graphical content appear seamless with the reality. Instead of just adding the graphical content as an overlay to the image more has to be done.

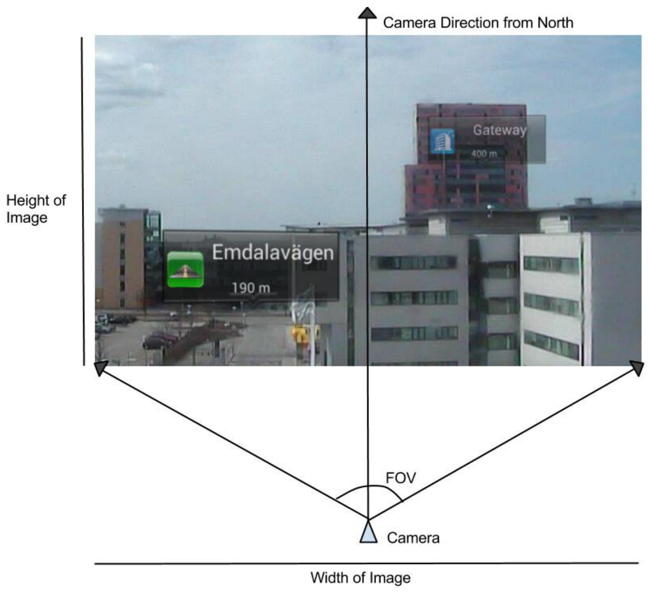


FIGURE 5.8: The mapping from the direction to its screen position.

Depth cues describes the way objects appear in reality in a cognitive way [28] . Objects can be further away from the starting point or be placed different, and this needs to be taken into account. These cues can be divided into two areas, monocular and binocular which corresponds to viewing the object with one or two eyes [28]. Since the cameras only uses one lens, only monocular depth cues can be used. To make the POI appear as real as possible will several depth cues be combined and applied on the graphics. When applying the depth cues to the image the human brain can just by viewing the object determine how far away it is, both relative to other POIs and to the environment.

In the examples below, the POI named *Gateway* is further away from the camera then the other POI named *Emdalavägen*. When applying different depth cues the cognitive matter of our brain sees and experiencing the *Gateway* as an object further away.

Occlusion is a monocular depth cue which describes that objects that is closer to us blocks the objects behind, as we can't see through physical objects. As illustrated in Figure 5.9 the closer POI is in front of the other.

When objects are further away it appears smaller the objects closer. **Relative size** is achieved just by changing the size of the object relative to the others, this can be seen in the Figure 5.10.

Relative height is accomplished by the way we humans use the horizontal line as a reference when determine how close the objects are. Objects closer to the horizon are

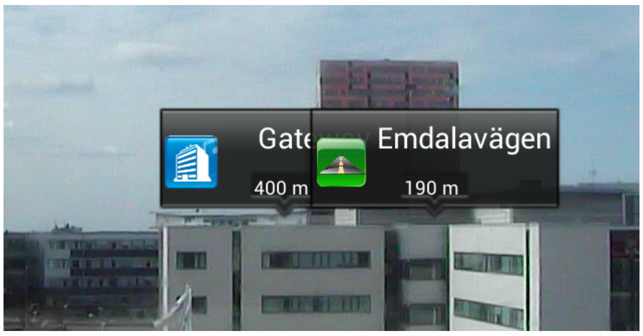


FIGURE 5.9: Depth cues with occlusion.

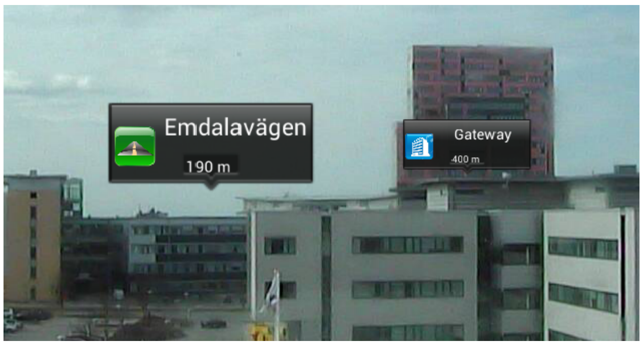


FIGURE 5.10: Depth cues with relative size.

further away. This can be compared to viewing the stakes of the bridge. See Figure 5.11 where one object is closer to the horizon than the other.

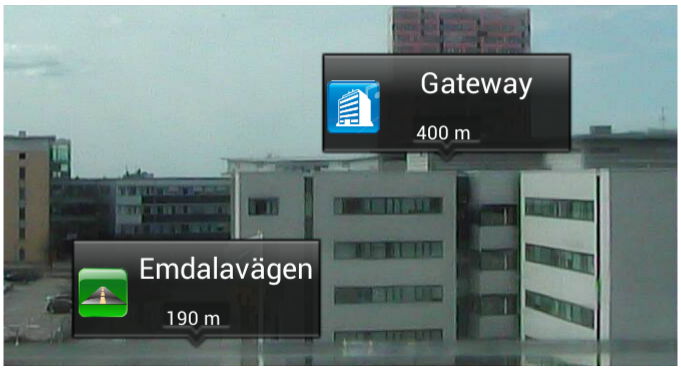


FIGURE 5.11: Depth cues with relative height.

Due to dust in the atmosphere and also limitations of the human eye, an object that is far away seems to be blurry and have less detail then closer. Giving an object some transparency to the background will achieve this effect which will be illustrated in Figure 5.12.

One depth cue alone may not give the desired effect but putting them all together will definitely fool our cognitive mind to believe it can measure the distance to objects. In

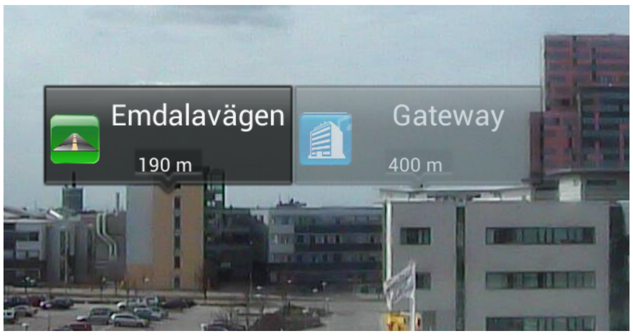


FIGURE 5.12: Depth cues with detail and atmospheric perspective.

the Figure 5.13 all the above depth cues have been implemented and displays the final result where everything is put together.

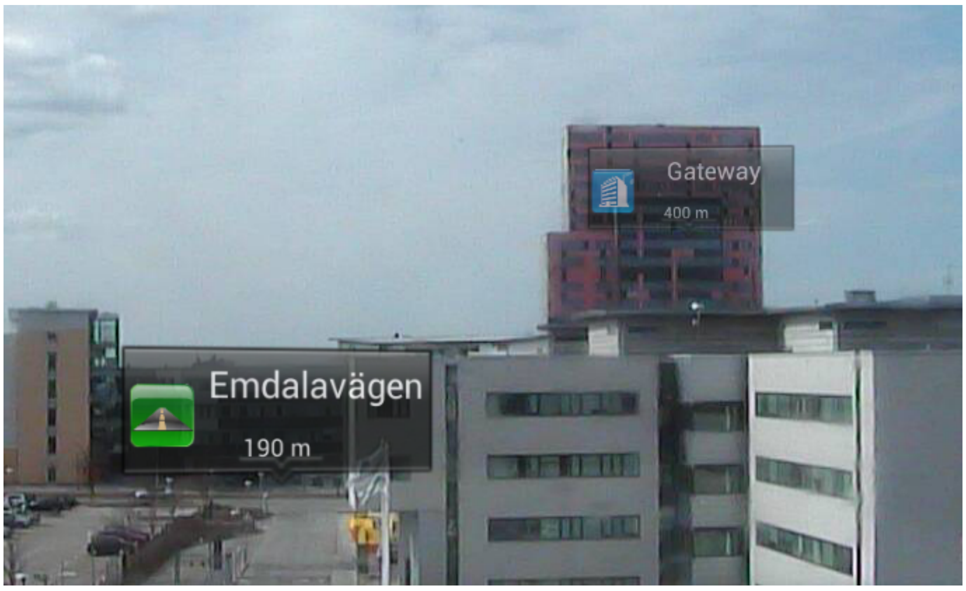


FIGURE 5.13: All the implemented depth cues together.

5.4.2 Camera Surfing

Camera surfing is the name and the outcome of a functionality in our AR-application. The main idea of our application is to simplify the overview over the cameras and what they can see, but also to make it fun to maneuver and navigate in the system. In the camera view other cameras can be seen along with the cameras POI, as can be seen in Figure 5.14. In this view other cameras is visible and clickable. When another camera is clicked on the view is translated to the new camera. Doing so, the user can surf through the world, from building to building or from country to country. It would also be possible to surf around between moving objects such as busses or airplanes.

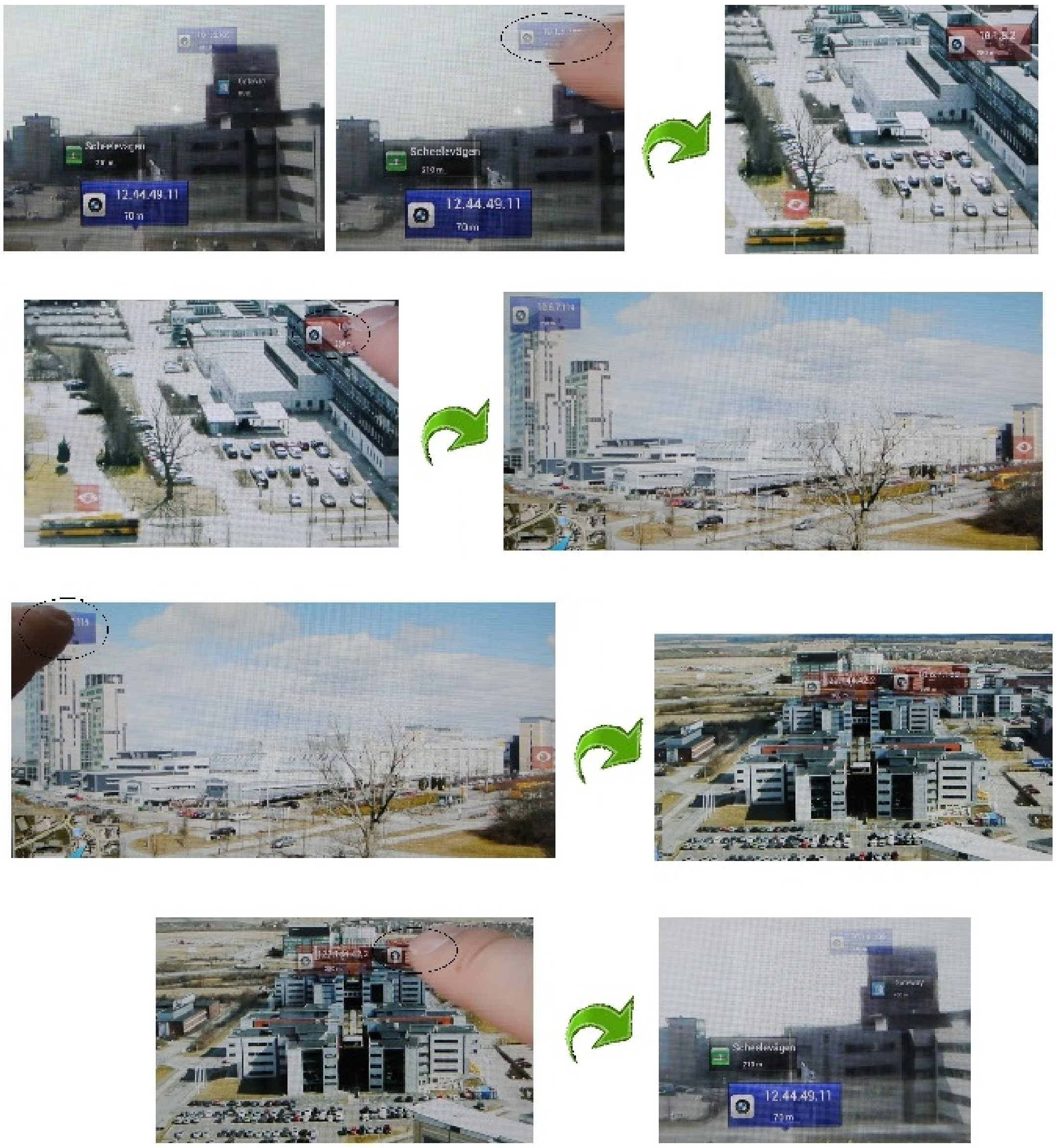


FIGURE 5.14: Camera surfing. Surfing in a loop starting from one camera back to the same camera.

5.4.3 Reverse GPS-positioning

The idea is to detect motion in the image and return the position of the motion. The position will be used to calculate the GPS-position of the object. This method will help the viewer to find and track objects or give information when an object is moving between different camera views. As a result of this technique the location of the object can be handed over between cameras.

The Reverse GPS-positioning method contains two steps. The first step is to get the screen position of the detected object and the second step is to calculate the depth in the image and calculate where to render the POI in the image.

5.4.3.1 Screen position

The first step of the motion detection is performed by looking at the previous and current frame followed by subtracting their pixel values. The difference in the subtraction and absolute value between the pixels values are then used to determine if there have been a movement. A threshold is then used to set the pixel color to black or white. This step is important later on so that the blob detection is able to determine if the pixel belongs to a blob or not. More on blob detection later in this chapter.

Equation 5.2 below subtracts the pixel values between two images. Each pixel is represented by three color values, the RGB values. When subtracting the images the red from the previous image is subtracted with the red from the current image. The same thing goes for the green and the blue color values.

I - Difference image/Threshold image

A - Previous frame

B - Current frame

$$I_{x,y} = \left|A_{x,y} - B_{x,y}\right| = \left|\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{x,1} & a_{x,2} & \cdots & a_{x,y} \end{pmatrix} - \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,n} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{x,1} & b_{x,2} & \cdots & b_{x,y} \end{pmatrix}\right| \tag{5.2}$$

When the subtraction is done the threshold t is compared with each pixel. If the pixel value is greater or below the threshold the pixel gets a new value as described in equation 5.3. Figure 5.15 shows the end result of the threshold image and Figure 5.16 shows the differences in red printed on the original image.

$$I_{x,y} = \begin{cases} \textit{black} & \text{if } I_x, y > t \\ \textit{white} & \text{if } I_x, y < t \end{cases} \tag{5.3}$$

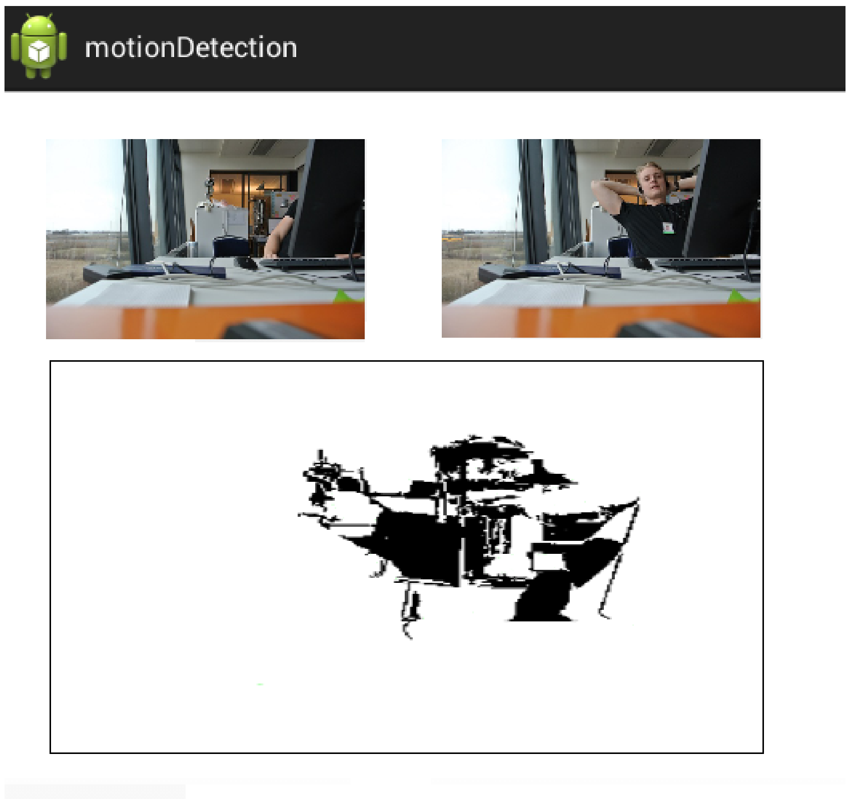


FIGURE 5.15: Difference between two frames marked with black or white.

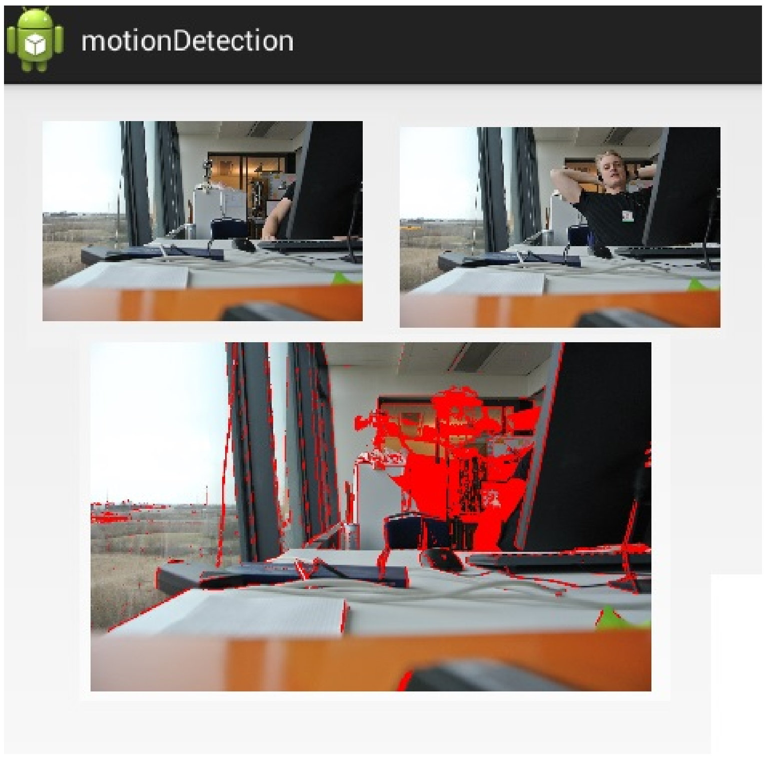


FIGURE 5.16: Difference between two frames with a slightly higher threshold marked with red pixels in the current image.

The next step is to get the position of the difference in the image. The naive way is to traverse through the image and collect connected pixels and add them to a specific blob, this blob will represent a group of connected pixels. When the blobs are saved they will have different size. The size of the blobs can now be sorted and a vector of the blobs that are interested in size will be returned. The blobs are returned as an area of the blob but in our implementation we needed a mass center from the blob area to be able to place the POI on the moving object. An open source blob detection was used called cvBlob [29]. The cvBlob library returns all blobs, so sorting them and finding the mass center was done before finding the GPS position, described in sub Section 5.4.3.3

The implementation that compares each two of the following frames and returns a blob area became jittery and returned a lot of movements in the image. And because it compares the two latest frames the motion detection can miss some parts. An example of when the motion detection fails is when two frames has parts of identical pixels in a moving object. The motion detection then fails because the motion detector says that the pixels are the same and does not trigger for motion. To come around this, the current image can be compared with a foreground image or reference image of the scene. Now every motion compared with the foreground is registered. This lead to the implementation of the reference image.

5.4.3.2 Reference image

The reference image technique was thought of and implemented during our work in the summer of 2012. The idea is to save one reference image; often the first image of a sequence of images captured by the camera and then to update the image continuously. The reference image is updated and the pixels that are detected to be changed is either increased or decreased with a step value. The step value changes how fast the reference image is updated.

R - Reference image

r1 - red pixel value of the current image

r2 - red pixel value of the reference image

Equation 5.4 describes the update of the red value. Same thing is done with the green and blue value of the pixel in the reference image. The r2 is then saved down together with the updated green and blue value to the reference image $I_{x,y}$.

$$r2 = \begin{cases} r2 + stepvalue & \text{if } r1 - r2 > 0 \\ r2 - stepvalue & \text{else} \end{cases} \tag{5.4}$$

In this way the image tend to filter out movements that may be caused by some object that stand still for a long while or if the step value is changed the object is first showed after an arbitrary of time. This could be used to remove cars in motion or people walking. In our implementation we wanted to remove noise or objects that triggered the motion detection during their movement to get a smooth and more stable motion detection that returned positive screen positions of the object.

5.4.3.3 GPS position and tracking

The next part is to use the screen-position of the motion detected and to determine the GPS-position. The new position can then be used by other cameras to know exactly where the object is. These cameras can use this position to rotate to the correct direction and also give a hint of where in the image the object should be. A GPS-position can also be used just to GPS-track different objects of their location to later see their movement over a map. Figure 5.17 illustrates an overview of the problem where the (x,y) values are the screen-positions and the (Latitude, Longitude) is the corresponding GPS-position.

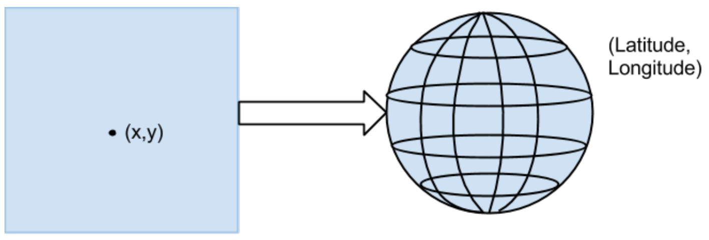


Figure 5.17: The process to go from screen-position to GPS-position.

The problem can be associated with the previous formula where two GPS-positions where known and the screen-position was calculated. Now the screen-position and one GPS-position is known and the other GPS-position is calculated.

To get the GPS-position of the moving object, three values need to be calculated. The first one is the GPS-origin, which in this case will be the camera spotting the motion. The GPS-origin is already known. The next two are not quite as easy as the first, *distance* and *direction* from the origin needs to be calculated.

Direction or bearing is determined by the fact that the direction of the center is known and the camera direction and the edge of the images is the cameras FOV. The ratio from the center of the camera to the edge can be mapped to the images width. By doing so, the direction from the camera and the moving object is calculated.

The *Distance* to the object is harder and needs some configuration to determine. There are several ways to get the distance to the object and one example is to use a IR-sensor, but for this project something more feasible is needed. One way to calculate the distance is to map the image positions to actual distance in reality. If an actual distances is known and it is known how far one pixel is in reality, an approximated distance can be calculated. In Figure 5.18 the process is displayed. The two boxes in blue are put in by the user during a configuration stage and the green box is later calculated by the program to determine the distance. The method works in a linear matter where the ground is plain and without buildings, but if something moves on top of a building the distance will be calculated wrong since it is not on the ground. It is however just an approximate distance so that the other cameras will have some clue on where the object is located.

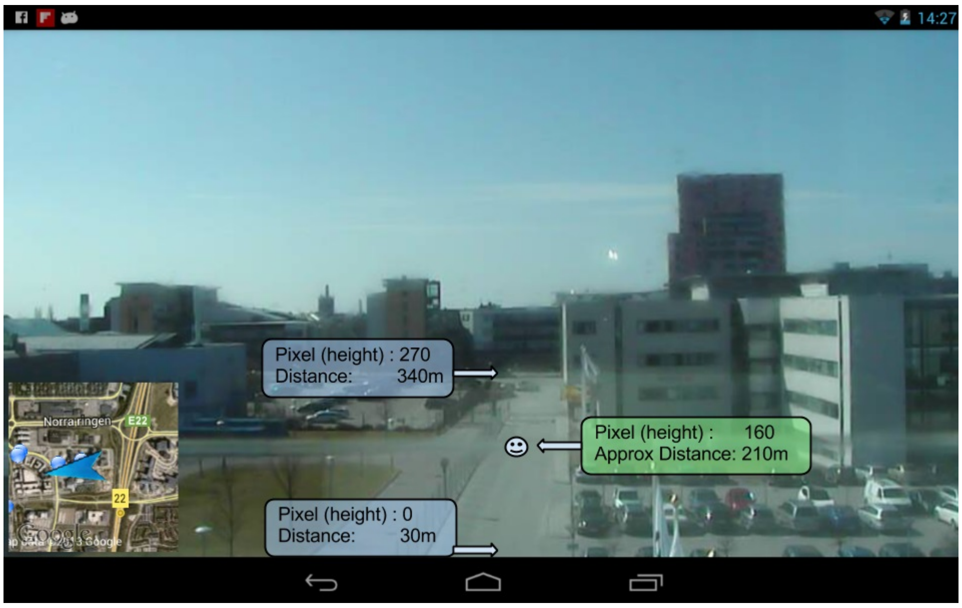


FIGURE 5.18: The process to go from screen-position to GPS-position.

With the three parameters the new GPS-position can be calculated. Since the earth and the GPS-coodinates follows a spheric shape it is not possible to see these positions as a 2-dimensional plane [30]. Instead the Formula 5.5 is used.

$$\omega_2 = \arcsin(\sin(\omega_1) * \cos(d/R) + \cos(\omega_1) * \sin(d/R) * \cos(\theta)) \tag{5.5}$$

$$\gamma_2 = \gamma_1 + atan2(\sin(\theta) * \sin(d/R) * \cos(\omega_1), \cos(d/R) \sin(\omega_1) * \sin(\omega_2)) \tag{5.6}$$

In the equations 5.5 and 5.6 *Omega* is latitude, *Gamma* is longitude, *Theta* is the direction or bearing, d is the distance away and R is the earth's radius (d/R is the angular distance, in radians).

A GPS-position for the moving object is now known and can be used by nearby cameras do target the area, in the Figure 5.19 another camera has detected an object which has been given a GPS-position which is displayed by the camera with a green POI. To make the observer notice the POI, the POI is blinking for a few seconds to draw attention.

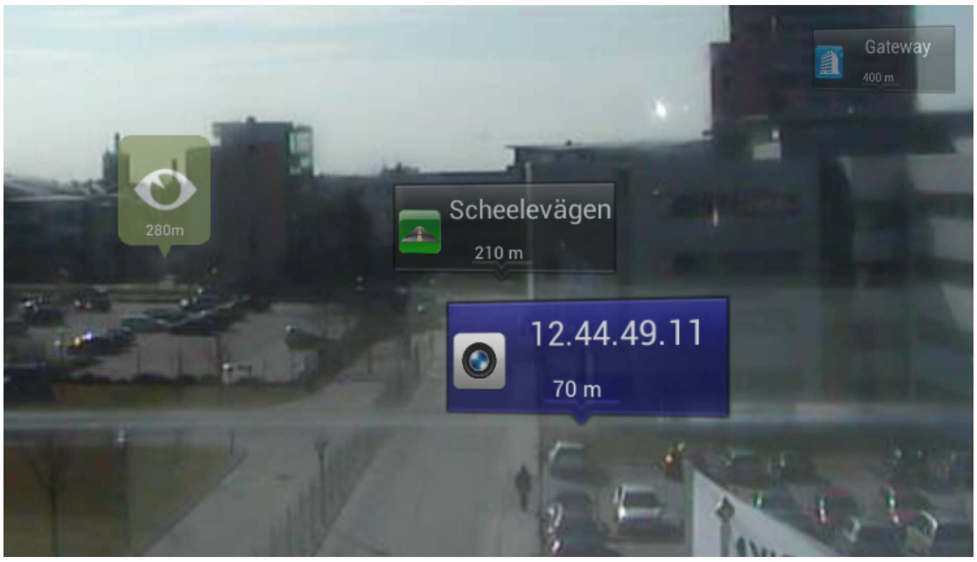


FIGURE 5.19: An object has been detected by another camera and displays its location in the current image.

Chapter 6

Results and Analysis

6.1 Result

The first part of the application, the Map View, was not intended and formulated in the goal of the thesis. It was a consequence of the different ideas of a desirable overview of the cameras and an solution to the configuration of the AR system. Regardless the formulation of the goal, the Map View became a big part of the application with many benefits itself. The visualization of the camera list greatly helps the orientation and also the navigation between different cameras. It is easy to add, edit and remove cameras with the map and to see which cameras viewing what. It is also possible to add own maps as an overlay and the whole system are highly configurable which for the customers is truly desirable. The comparison of the old system and the visualized is displayed in Figure 6.1.

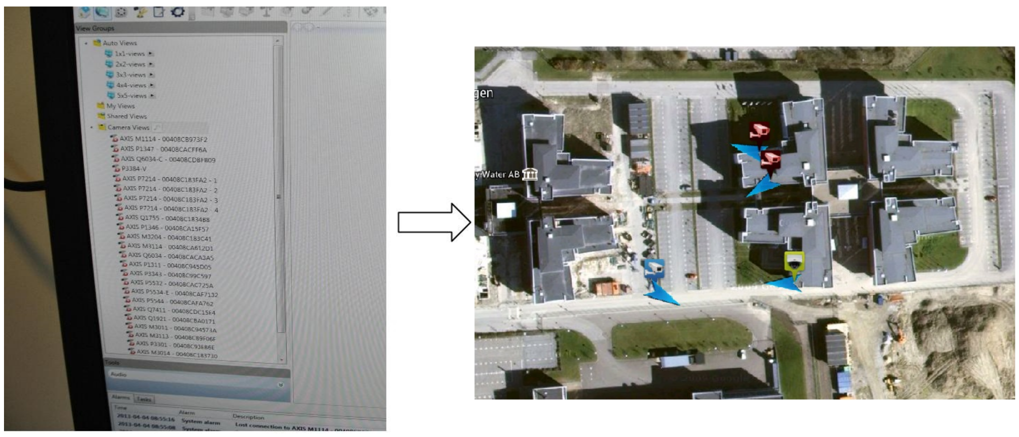

View Groups
Auto Views
1x1-views
2x2-views
3x3-views
4x4-views
5x5-views
My Views
Shared Views
Camera Views
AXIS M1114 - 00408CB973F2
AXIS P1347 - 00408CACFF6A
AXIS Q6034-C - 00408CDBFB09
P3384-V
AXIS P7214 - 00408C183FA2 - 1
AXIS P7214 - 00408C183FA2 - 2
AXIS P7214 - 00408C183FA2 - 3
AXIS P7214 - 00408C183FA2 - 4
AXIS Q1755 - 00408C1834B8
AXIS P1346 - 00408CA15F57
AXIS M3204 - 00408C183C41
AXIS M3114 - 00408CA612D1
AXIS Q6034 - 00408CACA3A5
AXIS P1311 - 00408C945D05
AXIS P3343 - 00408C99C597
AXIS P5532 - 00408CAC725A
AXIS P5534-E - 00408CAF7132
AXIS P5544 - 00408CAFA762
AXIS Q7411 - 00408CDC15F4
AXIS Q1921 - 00408CBA0171
AXIS M3011 - 00408C94573A
AXIS M3113 - 00408C89F06F
AXIS P3301 - 00408C93EB6E
AXIS M3014 - 00408C183730
Tools
Audio
Alarms
Tasks
Time
Alarm
Description
2013-04-04 08:55:16
System alarm
Lost connection to AXIS M111
2013-04-04 08:55:08
gen
Water AB

FIGURE 6.1: Axis Camera Station compared to Axis Maps

As mentioned earlier in Chapter 2.2, AR is rare and nearly nonexistent in the field of surveillance systems. At the beginning, only a diffuse idea about the benefits was the lead of the project and who led to the Camera View implementation. The Camera View was to be evaluated and tested in other conditions than clear and sunny weather. Even if AR increases the orientation in clear weather as well, it is not one of its greatest strength since all the buildings and roads are visible. The test was made in more demanding situations which would be weather with fog and also during night time. There are many other demanding situations for instance when the heat reaches high temperatures the thermal cameras tend to blur things out. Due to time restricted the test consists of to two tests. The result is viewed in Figure 6.2 and in Figure 6.3.

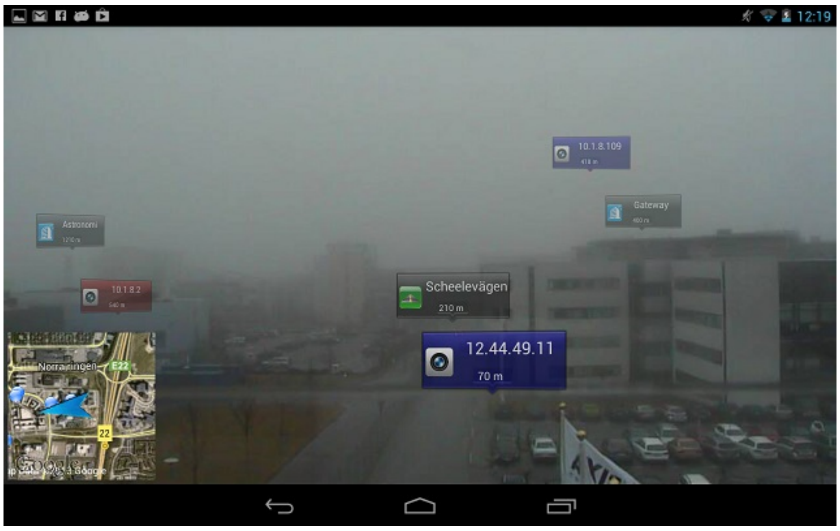


Figure 6.2: Orientation became much better in foggy environments

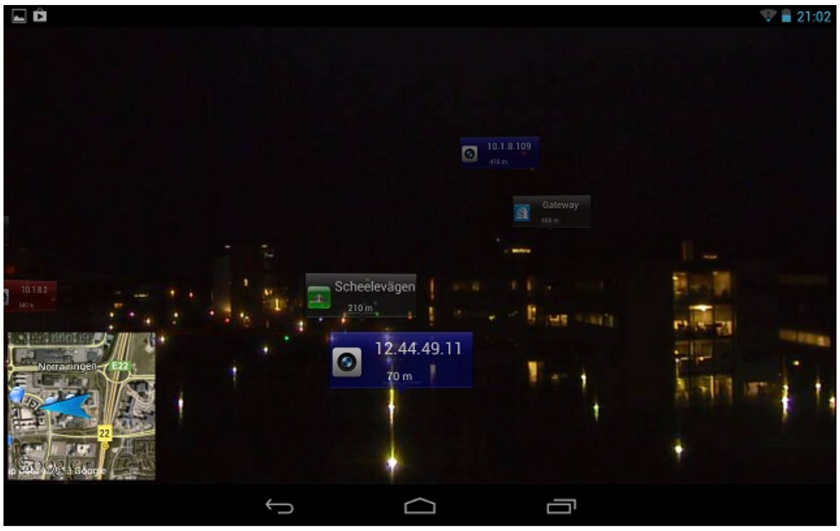


Figure 6.3: Orientation became much better during night time

As Figure 6.2 and Figure 6.3 indicates, it can be quite hard to separate structures from each other and the vision can be very limited. With the additional information in the image, you may not see more, but it is possible to know what you should see and to get an increased orientation. Figure 6.2 and Figure 6.3 only shows one camera, but with a full system the orientation and navigation gets even more complicated and the benefits with the AR system increases even more.

Using the cameras of Axis increases the precision of placing POIs on the correct location. The specific information about the FOV, which has a great effect on what is viewed in the scene is known for the Axis cameras. This contributes to a better POI positioning than with a smartphone camera, where the FOV is unknown. Without the FOV, the position of the POI is more of a guess rather then carefully calculated.

6.2 Demonstration video

A demonstration video was created to be able to view the concepts of the application. The video shows every element of the application and is configured with several cameras to view its strengths. The demo is not yet published online. But it is still possible to view the video or get some demonstration parts, by contacting one of the authors. If the video is being uploaded on YouTube in the future, try search for "Master thesis - Augmented Reality in Surveillance Systems".

6.3 SWOT Analysis

SWOT analysis is a method to evaluate the Strengths, Weaknesses, Opportunities, and Threats in a project [31]. The outcome of the analysis is an increased value for the customer which hopefully will improve the competitive advantage. The aim of the analysis is to turn the products weaknesses into strengths, and threats to opportunities and will finally give the manager options to match internal strengths with external opportunities. [32][33] An overview is viewed in the figure 6.4 to demonstrate the different sections.

Strengths

- A new way to interact with cameras

Helpful
to achieving the objective
Harmful
to achieving the objective
Internal origin
(attributes of the organization)
External origin
(attributes of the environment)
Strengths
Weaknesses
Opportunities
Threats

Figure 6.4: SWOT analysis with the different sections, illustration from Wikipedia

- The unique way to orientate and navigate
- Use of the external software *Google Maps* to ease integration
- Configure the system
- The external AR system which makes it possible to use todays cameras
- Low cost of production
- Higher operational efficiency
- New application of existing technology

Weaknesses

- Costs from usage of Google Maps
- The expensive computations which requires OpenGL and new hardware to the camera
- AR might appear as futuristic and not feasible in todays systems
- The system cannot be local but rather needs a server to run from
- New product needs to educate the market

Opportunities

- AR is yet unexplored in surveillance systems which makes the product unique
- No map systems with easy configuration on the market
- AR is a new way to enhance the image of an IP-camera

Threats

- The industries of AR is constantly growing and might conquer surveillance in the near future
- Google Maps is open for all companies including competitors
- Patents blocking the use
- Lack of demand

Chapter 7

Discussion and conclussions

7.1 Discussion

The work during the thesis was educational with plenty of self-development. We learned to approach new technologies and to sew together software with hardware APIs. The work got us to an end result where we could see the implementation come to life in a satisfied way. This was achieved in a numerous ways.

One interesting part of the thesis was the new way to use AR. The existing applications make use of a built in smartphone camera or of a web camera which differs from the stationary cameras of Axis Communications. Once configured and installed, the cameras will not change their location. This contributes to a better AR system then many others since the POIs can be adjusted to the correct height. Other applications uses the same height for all POIs and rather just points out t he direction which in fact is just an improved compass. With the stationary cameras, the scene can be configured and position the POI where the building or road actually are.

The design phase that included many brainstorming sessions, a couple of storyboards and conceptual ideas that were visualized led to the Lo-Fi prototype. The Lo-Fi prototype gave us important feedback from the test users and our ideas were improved and led to the final design and implementation of the Hi-Fi prototype. We could state that the first stages of design were very important for the final product since it gave great insight of different problems and areas. Without these sessions, the product would lack many important features and not to be of usage for the customer.

Another advantage is the use of http-requests to rotate the camera. When the user changes the direction, the application and the camera will have exactly the same values and they will be static. These will not be changed until the next time the user interacts with the system. The static values will give a static POI which means, it will not move. This differs from the cellphone where the sensor values changes all the time, even if the phone is still. In comparison, the static system will give a POI which is still and the cellphone will have a POI which moves and is rather shaky.

The approach of using our own AR system was very successful. This made it possible to use Axis own camera and integrate the cameras with a map system without spending too much time to learn and comprehend a existing AR API. When reflecting over the different tools to use for AR, this was the correct choice.

For the Map View, it is the other way around, to not use an API like Google Maps would be extremely time demanding and would probably not give any result due to the complexity. The Map View became very useful and gave us much appreciation. This was a bonus feature in our work. Since the aim was to examine the possibilities of using AR in a single Camera the way of give a complete overview of AR increased cameras in a system became powerful. When comparing with existing software that lists or handle multiple cameras, our Map View is easier to use and learn and gives the user a better overview and makes the configuration simple. The final conclusion is that Google Map was easy to use and integrate with the system and is a good choice for future products by Axis Communications.

7.2 Conclusions

The first step of the project was to investigate the area of AR and to come up with different ideas, both futuristic and suitable for today's systems. The different design sections gave many inputs to the matter and several ideas were presented as conceptual images, paper sketches and videos. Many of the ideas used HMDs as the AR-device which was intended for the future products. Since it was important to create an application which can be used today and view the strengths of AR to Axis Communications own cameras the Axis Maps application was formed.

The Axis Maps is intended for a large camera system where cameras easily can communicate and share information such as a spotted intruder. The AR part highly increased the orientation with the POIs and also makes the navigation between cameras much easier. The POIs also adds more information such as distance to the image but can also be used as a privacy mask.

Since the AR-tracking is used with location-based tracking the computations is suitable for Axis own cameras and no image analysis is needed.

A list of all the implemented and not implemented features is listed in Appendix A. Most of the desired features are implemented which makes the application fully usable and demonstrates many positive angles of AR.

7.3 Future work

Much has been done but there are much left. The application is based on scenarios, brainstorming and storyboards. Since no surveys were done, the actual customers will have many additional inputs and reflections before it is suitable for their systems. In this thesis we focused on just a few features where many are left to implement to have a full system. The area of AR is full of potential and the opportunities are many, even more research would lead to more ideas and in the end, a better product.

Since the application is just a prototype of a product, a full implementation and integration with Axis Communications own system has to be done which supports several systems rather than just one. Such a system would be a cloud-solution which Axis would provide for the customers.

In the beginning of this master thesis during our many brainstorming sessions, many somehow futuristic ideas was thought of and in the final implementation traces of almost all the ideas was to be found. Even though these ideas weren't implemented on its own, they should be accounted for in the future when devices like HMDs are more known and used.

Appendix A

Requirements and Functionality

A.1 Necessary

These requirements needs to be implemented to have a working and an acceptable application.

- **R1. Add a new POI** *Implemented*

  The user should be able to add POIs and insert information in form of text.

- **R2. Delete a POI** *Implemented*

  The user should be able to delete a selected POI.

- **R3. Edit a POI** *Implemented*

  The user should be able to edit a selected POI.

- **R4. Select other cameras that's viewing the POI** *Implemented*

  If several cameras have the same POI, the user should be able to click on that POI and get a list of the cameras that also sees it.

- **R5. Add a new camera** *Implemented*

  The user should be able to add a new camera.

- **R6. Delete a camera** *Implemented*

  The user should be able to delete a selected camera.

- **R7. Map overview** *Implemented*

  The user should be able to see an overview of the cameras positioned on the map. Either directly on the screen or in some menu.

- **R8. Text in image** *Implemented*

  The user should be able to view the surrounded points as text in the video on the correct locations.

- **R9. Search on street or building** *Not implemented*

  The user should be able to search for a specific street or building and get a list for the cameras which are viewing the area.

- **R10. Rotate camera** *Implemented*

  The user should be able to rotate the camera by either swiping or to use other interaction methods.

- **R11. List the viewed buildings** *Not implemented*

  The user should be able to list the buildings and streets which is viewed by a camera

- **R12. Enter GPS coordinates to a camera** *Implemented*

  Since there are no embedded GPS system the user should be able to enter manually GPS coordinates to a camera.

- **R13. See POIs from the camera view** *Implemented*

  This will be the Augmented Reality part of the application, from the camera view, POIs will be put into (or upon) the image as they appear in the real world.

- **R14. Change Altitude of POI** *Implemented*

  The user should be able to change the altitude of any POI from the Camera View.

- **R15. Depth cues for POI** *Implemented*

  To make the POI look more real, there should be some depth cues applied to it.

- **R16. Distance to POI from camera** *Implemented*

  Every POI should have a field which contains the distance from the camera, and it should be displayed on the billboard.

- **R17. Choose categories in Map View** *Implemented*
  From the map view, the user should be able to choose which cameras to view on the map

- **R18. Choose nearby cameras viewing specific area** *Implemented*
  The user would only have to press on the map and then receive the nearby cameras watching the area and be able to connect to that camera. This would simplify the case where the user don't know which camera viewing the different areas but rather just want any camera viewing the specific area.

- **R19. Get a GPS position from moving objects** *Implemented*
  In the camera view the application would reverse the POI placement. Instead of placing a POI on a specific pixel the application would take a moving object on a pixel and then track it to a GPS position.

- **R20. Notify other cameras of moving objects** *Implemented*
  With the GPS position of a moving object the other cameras should be able to see it. It would be like the cameras notify each other of important events.

- **R21. Support buildings** *Implemented*
  To be able to support buildings, the user should be able to choose which floor to view and then view a plan instead of the Google Map of the specific area.

A.2 Desirable

These requirements are desirable for the product but don't have to be implemented to have a pleasing result.

- **R22. Choose which POI categories to see** *Not implemented*
  From the camera view, the user should be able to select which POI categories to see.

- **R23. Automatic road and building collection** *Not implemented*
  The system should automaticly collect nearby roads and buildings and add them to the list.

- **R24. List all cameras** *Not implemented*

  The user should be able to list all the cameras and see where they are positioned and on what address.

- **R25. Rotate text** *Implemented*

  Rotate text in the image due to the cameras field of view. If the text is far away from center it will rotate more.

- **R26. Resizable text** *Implemented*

  The distance from the camera to the object will affect the size of the text on the image.

- **R27. Small window** *Implemented*

  Instead of using 3D text in the image, a small window will be displayed instead with text inside it.

- **R28. Gesture based camera movement** *Implemented*

  Instead of using buttons, the user uses gestures to pan, tilt or zoom the camera.

- **R29. Street View pictures in POI** *Not implemented*

  Instead of using images on the POI for building, road etc. the application will collect images from Google Street View over the objective which will be much more informative.

- **R30. Moving cameras** *Not implemented*

  To follow moving cameras on for example busses and helicopters the system should support moving cameras with new GPS positions. The GPS position would not be static but rather constantly updated.

- **R31. Use FOV instead of arrow** *Not implemented*

  Instead of using an arrow to show which direction the camera is viewing, the FOV would be used which also displays the area which the camera can view with its FOV.

- **R32. Improved nearby camera** *Not implemented*

  Not only use the distance when determine which three cameras to display in the nearby camera algorithm, the cameras FOV and direction would also be accounted for, so that only cameras viewing the area are displayed.

A.3 Unnecessary

The following requirements have the least priority for the project, only if there's time these will be implemented.

- **R33. Placing own texts in the video** *Not implemented*

  The user should be able to place their own texts in the video by clicking on the screen which will be given correct coordinates and positions.

- **R34. Panorama images** *Not implemented*

  When cameras are close to each other, the application should support panorama images from the different cameras.

Bibliography

[1] Axis Communications AB. (June 2013). URL http://www.axis.com/.

[2] GPU. (June 2013). URL http://sv.wikipedia.org/wiki/GPU_(datorteknik).

[3] OpenGL. (June 2013). URL http://sv.wikipedia.org/wiki/OpenGL.

[4] Furth B. *Handbook of Augmented Reality. New York, Springer*, (2011).

[5] Svensson B. Wozniak M. *Augmented Reality Visor Concept. Lund, Department of Design Science*, (2011). URL http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=1894573&fileOId=1894587.

[6] Boström M. *Design and Implementation of a Mobile Application for Public Transport. Lund, Department of Design Science*, (2012).

[7] WikiTude. (April 2013). URL http://www.wikitude.com/.

[8] Ray-Ban Virtual Mirror. (April 2013). URL http://www.ray-ban.com/usa/science/virtual-mirror.

[9] Steve Mann. (June 2013). URL http://en.wikipedia.org/wiki/Steve_Mann.

[10] Google Project Glass. (Januari 2013). URL http://en.wikipedia.org/wiki/Project_Glass.

[11] Azuma R. *A Survey of Augmented Reality.* (1997). URL http://www.eat.lth.se/fileadmin/certec/ar_survey.pdf.

[12] McMahan A. *A Method for Analyzing 3-D VideoGames.* (2003). URL http://people.ict.usc.edu/~morie/SupplementalReadings/ch3-McMahanrev.pdf.

[13] Wc3Schools.com. (Januari 2013). URL http://www.w3schools.com/browsers/browsers_display.asp.

[14] Gestures. (May 2013). URL http://en.wikipedia.org/wiki/Gesture.

[15] Natural Language Interface. (May 2013). URL http://www.artificial-solutions.com/natural-language-interaction/.

[16] Axis Communications Hardware. (June 2013). URL http://developer.axis.com/wiki/doku.php?id=axis:hardware.

[17] Axis Communications Thermal Camera. (June 2013). URL http://www.axis.com/products/video/camera/thermal/index.htm.

[18] Balsamic Mockups. (March 2013). URL http://www.balsamiq.com/products/mockups.

[19] Microsoft Sketch Flow. March 2013. URL http://www.microsoft.com/silverlight/sketchflow/.

[20] Heinen S. *DroidAR*. (March 2013). URL https://github.com/bitstars/droidar.

[21] GNU GPL v3. (March 2013). URL http://www.gnu.org/licenses/gpl.html.

[22] Metaio. (March 2013). URL http://www.metaio.com.

[23] Metaio Awards. (March 2013). URL http://www.metaio.com/company/.

[24] Mixare. (March 2013). URL http://www.mixare.org/.

[25] Gmap. (March 2013). URL http://greatmaps.codeplex.com/.

[26] Google Maps API. (March 2013). URL http://developers.google.com/maps/.

[27] VAPIX. (April 2013). URL http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php.

[28] VR Compendium. (June 2013). URL http://www.eat.lth.se/fileadmin/eat/MAM101/VR-Kompendium2013.pdfL.

[29] Blob Detection Android. (March 2011). URL https://code.google.com/p/cvblob-for-android/.

[30] Formula to calculate new GPS-position. (April 2013). URL http://www.movable-type.co.uk/scripts/latlong.html.

[31] Pahl N. and Richter A. *Swot Analysis - Idea, Methodology and a Practical Approach. Norderstedt, Germany, Books on Demand GmbH*, (2007).

[32] Wikipedia SWOT analysis. (April 2013). URL `http://en.wikipedia.org/wiki/SWOT_analysis`.

[33] Marketing Teacher SWOT analysis. (April 2013). URL `http://www.marketingteacher.com/wordpress/swot-analysis/`.