MASTER THESIS IN HUMAN COMPUTER INTERACTION AND DESIGN, SECOND LEVEL
*STOCKHOLM, SWEDEN 2019*

# Exploring Augmented Reality for enhancing ADAS and Remote Driving through 5G

*Study of applying augmented reality to improve safety in ADAS and remote driving use cases*

MAX JAN MEIJER

**KTH ROYAL INSTITUTE OF TECHNOLOGY**
***ELECTRICAL ENGINEERING AND COMPUTER SCIENCE***

# Exploring Augmented Reality for enhancing ADAS and Remote Driving through 5G

*Study of applying augmented reality to improve safety in ADAS and remote driving use cases*

Max Jan Meijer

2020-04-01

Second Level

Examiner
Konrad Tollmar

Supervisor
Pietro Lungaro

Industrial adviser
Stefano Sorrentino

KTH Royal Institute of Technology
School of Electrical Engineering and Computer Science (EECS)
Mobile Service Lab
SE-100 44 Stockholm, Sweden

# Abstract

This thesis consists of two projects focusing on how 5G can be used to make vehicles safer. The first project focuses on conceptualizing near-future use cases of how Advanced Driver Assistance Systems (ADAS) can be enhanced through 5G technology. Four concepts were developed in collaboration with various industry partners. These concepts were successfully demonstrated in a proof-of-concept at the 5G Automotive Association (5GAA) "The 5G Path of Vehicle-to-Everything Communication: From Local to Global" conference in Turin, Italy. This proof-of-concept was the world's first demonstration of such a system. The second project focuses on a futuristic use case, namely remote operation of semi-autonomous vehicles (sAVs). As part of this work, it was explored if augmented reality (AR) can be used to warn remote operators of dangerous events. It was explored if such augmentations can be used to compensate during critical events. These events are defined as occurrences in which the network conditions are suboptimal, and information provided to the operator is limited. To evaluate this, a simulator environment was developed that uses eye-tracking technology to study the impact of such scenarios through user studies. The simulator establishes an extendable platform for future work. Through experiments, it was found that AR can be beneficial in spotting danger. However, it can also be used to directly affect the scanning patterns at which the operator views the scene and directly affect their visual scanning behavior.

### Keywords

Augmented-reality, 5G, Advanced driver-assistance systems, Vehicle-to-vehicle, Vehicle-to-everything, Remote-driving, Eye-tracking, Internet-of-Things, Gaze

# Sammanfattning

Denna avhandling består av två projekt med fokus på hur 5G kan användas för att göra fordon säkrare. Det första projektet fokuserar på att konceptualisera användningsfall i närmaste framtid av hur Advanced Driver Assistance Systems (ADAS) kan förbättras genom 5G-teknik. Fyra koncept utvecklades i samarbete med olika branschpartner. Dessa koncept demonstrerade i ett proof-of-concept på 5G Automotive Association (5GAA) “5G Path of Vehicle to to Everything Communication: From Local to Global” -konferensen i Turin, Italien. Detta bevis-of-concept var världens första demonstration av ett sådant system. Det andra projektet fokuserar på ett långt futuristiskt användningsfall, nämligen fjärrstyrning av semi-autonoma fordon (sAVs). Som en del av detta arbete undersöktes det om augmented reality (AR) kan användas för att varna fjärroperatörer om farliga händelser. Det undersöktes om sådana förstärkningar kan användas för att kompensera under kritiska händelser. Dessa händelser definieras som händelser där nätverksförhållandena är suboptimala och information som tillhandahålls till operatören är begränsad. För att utvärdera detta utvecklades en simulatormiljö som använder ögonspårningsteknologi för att studera effekterna av sådana scenarier genom en användarstudie. Simulatorn bildar en utdragbar plattform för framtida arbete. Genom experiment fann man att AR kan vara fördelaktigt när det gäller att upptäcka fara. Men det kan också användas för att direkt påverka skanningsmönstret där operatören tittar på scenen och direkt påverka deras visuella skanningsbeteende.

### Nyckelord

Augmented-reality, 5G, Avancerade förarassistanssystem, Fordon-till-fordon, Fordon-till-allt, Fjärrkörning, Ögespårning, Augmented-reality, Internet-of-Things

## Acknowledgements

I want to start by thanking Pietro for being a great supervisor during my thesis. For both pieces of work will be discussed, your supervision has been very constructive, and it was a lot of fun. Additionally, I would also like to thank Konrad for being my examiner, as well as his advice on the directions of both projects and providing support for hosting the user studies.

I want to thank Stefano Sorrentino for being my industrial advisor on this thesis, as well as involving me in the realisation of the demo in Turin that we have all contributed to. It was a unique experience to be involved in this project. Finally, I would like to say thanks to Smriti Gopinath and Thorsten Lohmar for their collaboration building up to this event.

Cambridge, March 2020
Max Jan Meijer

# Table of contents

# List of Figures

# List of acronyms and abbreviations

| | |
|---|---|
| ADAS | Advanced Driver Assistance System |
| API | Application Programming Interface |
| AR | Augmented Reality |
| CNN | Convolutional Neural Network |
| FOV | Field of View |
| Glasses | Tobii Pro Glasses 2 |
| HCI | Human Computer Interaction |
| HMD | Head Mounted Display |
| HMI | Human Machine Interface |
| ML | Machine Learning |
| IoT | Internet of Things |
| UI | User Interface |
| UX | User Experience |
| VR | Virtual Reality |
| V2V | Vehicle-to-Vehicle |
| V2X | Vehicle-to-Everything |
| YOLO | You-Only-Look-Once |

# 1 Introduction

The introduction of 5G has great potential for many stakeholders within the transportation domain. It can support a lot of communication types that enable scores of opportunities for new use cases to make traffic safer, faster, and more sustainable. Two of the most critical areas include vehicle-to-vehicle (V2V), where vehicles relay signals to each other, and vehicle-to-everything (V2X), where vehicles communicate with any potential artifact that has sensors, such as traffic lights and smartphones [1, 2].

This thesis focuses on two threads of work. Both explore how 5G can contribute to increasing safety in operating vehicles. One part focuses on systems that aid the individual to drive more safely and avoid accidents, also known as ADAS. Here, the focus is on applying augmented reality inside of the vehicle itself. The other focuses on future use cases in which semi-autonomous vehicles are being remotely monitored from a control tower. Here, the focus is on applying augmented reality to aid the operator in preventing dangerous events. Therefore, the augmented reality is used as a 'remote ADAS' to improve safety in the remote operating of vehicles.

This chapter covers the specific challenges that are addressed within the thesis, their context, as well as the goals and structure of the thesis.

## 1.1 Vehicle Autonomy

A self-driving car, also referred to as an autonomous vehicle (AV), is a car able to drive itself safely through its environment, with little to no human input or corrections [3]. Exciting recent developments in fields such as computer vision and machine learning have resulted in a massive surge of attention for autonomous vehicles in the last few years. However, at this very moment, a commercially available and fully self-driving vehicle hasn't become a reality yet. Although some vehicles are marketed as self-driving, such as the Tesla Model S [4], these cars can be considered operating at a "level-2" out of the 5-level scale as defined by The American Highway Traffic Safety Administration (NHTSA) [5]. These levels can be summarized as follows:

Level 0 - No Automation: the driver performs all the tasks;

Level 1 - Driver Assistance: the driver controls the vehicle, but some automated features assist the driver (such as cruise control);

Level 2 - Partial Automation: vehicle has combined functions that are automated, such as acceleration and steering, but the driver needs to be continuously engaged during operation;

Level 3 - Conditional Automation: the driver is still a necessity, but the driver does not require to monitor the environment until notified;

Level 4 - High Automation: under certain conditions, the vehicle is capable of performing all driving functions. The driver may still take control of the vehicle;

Level 5 - Full Automation: the vehicle is capable of performing all driving functionalities under every condition.

Modern vehicles currently provide partially automated features. These technologies are referred to as ADAS (Advanced driver-assistance systems). They are electronic systems that assist the driver during the operation of the vehicle through automation, adaption, or enhancements for improving vehicle safety and driving experiences [6]. New functionalities of ADAS implementations include technologies such as cruise control and parking assistance. The systems that are on the market today mostly focus on sensors that are present in the car itself. By using input from multiple data sources such as radar, LIDAR (similar to radar, but uses light), camera, and ultrasound, ADAS

systems can get a (basic) understanding of the world around the vehicle. However, these systems are currently limited by what sensors onboard of the vehicle can measure by themselves.

Next-generation ADAS are likely to leverage the capabilities of 5G wireless connectivity to enable use cases in which sensor-data in between other vehicles (V2V - vehicle-to-vehicle) and traffic infrastructure and pedestrians (V2X - vehicle-to-everything) are being shared [7]. A big advantage of such data-exchange is that other participants in traffic can be made aware by third-parties' detections;

## 1.2 5G Technology

5G is the fifth generation of wireless technology for digital cellular networks. Before the technology is discussed in further details, and its impact on the automotive industry, the following provides short summary of what previous generations have enabled:

1G: Mobile Voice calls

2G: Mobile Voice calls and SMS

3G: Mobile web browsing

4G: Mobile video consumption and higher data speeds

Comparing 5G to 4G will bring the following: 100 times faster data rates, significantly reduced latency (1-10ms compared to 40-50ms) and the ability to dedicate part of the 5G network to a specific service, also referred to as network slicing [8].

For vehicles, a 5G network brings excellent benefits when it comes to communicating with other traffic participants. Although, vehicles on today's 4G network can already broadcast information such as location, speed, and direction. 5G opens the door to many time-critical use cases as well as use cases that require more data to be streamed between parties than current 4G networks can handle. An example of a time-critical use case is when vehicles need to negotiate whose turn it is to cross at an intersection between sAVs (Semi-Autonomous Vehicles) as they approach the crossing. An example of a use case in which both increased data streaming and lower latency become essential is related to remote monitoring or controlling of sAVs. In the case of remote driving, the vehicle is not fully autonomous. Instead, it relies on cooperation with a remote operator that can take control of the vehicle when necessary. The vehicle can also be considered to be a cooperative vehicle, rather than being an autonomous vehicle; given that it will still rely on the human-in-the-loop.

There exists one major challenge for these types of use cases. This lies in probably the biggest shortcoming of 5G technology: it's range of operation is very small compared to its previous generations. A 5G cell can serve cellular data within a range of about 250 meters in optimal conditions [9]. By comparison 4G wavelengths have a range of about 10 miles [10]. Another issue on top wavelength is that the signals of 5G can be easily hampered by physical obstacles due to their shorter wavelengths. Small objects such as leaves on a tree or natural events such as rain can already decrease the effective range of a 5G cell. Another factor that has to do with the range of a cell is handovers. These handovers occur whenever a cellular device moves 'closer' from one base station to another. During the handover, there is a time in which no 5G connectivity will be sent to the receiver. However, they may still receive 4G or 3G during that period. As a result, however, the amount of uplink the vehicle has available is much lower, and the data that can be sent in an amount of time is significantly reduced. If a cooperative vehicle sends a video-stream to its controller, then this video stream will be sharply reduced in quality. An example of this can be seen in figure 1.

What makes such events dangerous is that critical details about traffic can disappear as a result of these glitches when the remote operator relies on a video stream to asses the situation at hand. Pedestrians, for example, can disappear completely from sight and be at risk as a result of these handover situations.


Case 1: Two pedestrians in the distance disappear as the vehicle is approaching them


Case 2: A pedestrian distance disappears as the vehicle is approaching while making a turn to the right

**Figure 1: Two situations that illustrate the dangerous effects of a base-station handover and how it reduces the quality of the video-stream temporary.**

For any use case that is enabled through 5G it is essential to keep these factors in mind. The challenge is that many systems always need to remain operational, even when the network quality may not be up to the “5G standard” that they are designed for. What this means, is that these systems need to have possible fall back solutions. For example, for remote monitoring or operation of vehicles, this means that the system should provide additional fall back mechanisms. These mechanisms should compensate for events when data that is being streamed - such as the video - is in very poor quality.

### 1.3 About both projects

As discussed in the previous section, 5G technology has a great potential for automotive use cases. However, there are also big challenges to overcome. In this thesis, two areas of work will be explored:

1) One focus area of this thesis will explore the potential of 5G for ADAS use cases. The challenge of this work is to build the world's first 5G enhanced ADAS system. This is done through the creation of a proof-of-concept of various ADAS implementations on a live and operational 5G network. This work is done in collaboration with KTH and Ericsson, including various partnering companies that will be introduced later. The goal is to present this system during the 2019 5GAA Conference in Torino, Italy [11]. From now on, this work will be referred to as the "Torino Demo".

2) The other area of the thesis investigates one of the significant challenges for remote monitoring and driving of vehicles. Namely, creating a fall back solution that augments the video stream from a remotely operated vehicle if the network doesn't cover an area or is performing poorly due to other unexpected events. The aim is to compensate for poor video streaming quality using AR (augmented reality), to improve the capabilities of the operator/observer to recognize possible dangerous events to maintain the safety of operation.

The overarching research questions are related for both parts of the thesis:

1) How can we develop a proof-of-concept that demonstrates ADAS enhanced through 5G?
2) Can AR be used to improve the ability of a remote observer of a vehicle to detect dangerous events during critical moments in which the data stream to them is limited?

Based on theory and background, both these questions will be defined more concretely in the following chapters.

## 1.4 Purpose

Both parts covered in this thesis aim to contribute to their own individual purpose. This is in line with our vision on how futuristic some concepts may be. The integration of ADAS concepts using 5G technology is less futuristic than remotely monitoring sAVs. The aim of the work on the ADAS is therefore focused on what can be achieved in the next product-cycle. Whereas the work on the remote driving scenario is focused on the a more distant timeframe.

For the Torino Demo, the purpose is to demonstrate various implementations of ADAS enhanced through 5G. Within this demonstration, both V2V and V2X use cases will be integrated and shown as part of a cohesive scenario. So far, a working implementation of such a system hasn't been demonstrated before. As part of this purpose, it demonstrates the capabilities of how safety in vehicles can be further improved through these technologies. Additionally, it provides a glimpse of near-future possibilities of commercial systems. Furthermore, at the time of writing, many countries are still introducing their 5G infrastructure [12]. Having a 5G network is essential for both remote driving, as for V2V and V2X ADAS systems. Although it is a necessary step, policy around vehicle autonomy needs to be expanded as well [13]. Demonstrating its capabilities at events such as the 5GAA conference - which many policy makers and journalists attend - can aid facilitating these discussions.

Currently, research is ongoing into building systems that enable remote driving. One example of such efforts is current work being done by ITRL (Integrated Transport Research Lab) [14] in Stockholm, Sweden. This specific use case is therefore farther away into the future before it will be rolled out commercially. As part of this work, the aim is to build a simulator that can be used to evaluate the AR interactions through eye-tracking. As part of this experiment, other actors in traffic, such as pedestrians and vehicles, will be augmented to aid in their detection. The simulator will be capable of playing video fragments and apply augmentations to the video in real-time. By making use of eye-tracking technology, the AR interaction can be evaluated through a user study. The simulator will be developed in such a way that it can be easily extended, meaning that other videos, as well as AR-interaction(s) can be added in future work. Furthermore, by focusing on technology that is already available for the detection of such actors from video images, the implementation can also be handed over to ITRL should it prove to be successful.

## 1.5 Goals

For the Torino Demo, the goal is to contribute to the creation of a proof-of-concept demonstration of 5G enhanced ADAS services. This project contributes to the world's first realization of such a technical demonstration. Achieving this can aid in opening the discussion around policy for 5G and vehicle autonomy. Furthermore, it also provides a platform to showcase upcoming technology, which could help with marketing and commercialization efforts.

For the work on the monitoring of remote vehicles, the goal is to design and evaluate AR-interactions on their effectiveness for aiding in spotting dangerous events. An example of such an event is when the network cannot deliver a video stream to the control tower in an adequate quality. The work aims to investigate if AR can be used to compensate for such events, and thereby improve the safety of operation.

## 1.6 Method

For the work on the Torino Demo, the main contribution is the development of an HMI (Human Media Interface) for the inside of the vehicle, as well as ideation and development of various V2V and V2X use cases. These efforts are done in collaboration with KTH, Ericsson, and partnering companies (Audi, Qualcomm, TIM, Pirelli, Italdesign, and Tobii).

To test the designs, they are built into a functional prototype, which can be evaluated using the simulator. The user test involves participants observing a vehicle as its driving through a city. For a section of the test, video stream quality will be poor, while augmentations will be applied. Eye-tracking technology will be used to analyse how both lower video quality and augmentations, can affect the user's ability and behaviour during the test.

## 1.7 Structure of the thesis

Given that this thesis encompasses the documentation of two projects, this document has been split up into three parts. The first part is what you, as a reader, are reading right now. It aims to introduce the thesis as well as cover common concepts that both parts of the work share. The second part of the thesis aims to cover the work and research that has been done through the exploration of the effects of AR for remote observation of vehicles and the development of the simulator environment. Finally, the third part of the thesis covers work that has been done on the Torino Demo, World’s-first demonstration of 5G enhanced ADAS. This structure generally reflects the order in which work has been done during the thesis as well.

# 2 Background

This chapter provides background information about technology and techniques used within this project, as well as related work on automotive projects. Topics covered include augmented reality, object detection, and eye-tracking technology.

## 2.1 Augmented Reality

Augmented Reality (AR) is a technology that enhances objects that reside in the real world by adding computer-generated perceptual information to it. Although visual applications of AR are probably the most common, it should be noted that other sensory modalities can be included as well, such as sound and smell [15]. At its core, AR can be defined as a system that fulfills three features: a combination of real and virtual worlds, real-time interaction, and accurate 3D registration of both virtual and objects in the physical world [35]. In 1992, the term "Augmented Reality" was first used [16]. This is when the first embodiment of AR was created in a head-mounted display (HMD). This device provided virtual information based on the position of the head in real-time.


**Figure 2: A render of a Heads Up Display found in modern Audi vehicles. Image obtained from [17]**

Today, AR has also found its commercial applications within vehicles but not in the form of a wearable device, yet. Instead, the automotive industry has mostly been focused on developing alternative AR solutions. An example of an application is a Head-Up Display (HUD) by Audi, as can be seen in figure 2. This functionality displays information such as the current speed at which the vehicle is driving, as well as warnings about traffic situations (e.g., crossing pedestrians in the dark) [17]. Although this application is a relatively small and modest AR interface, there are also futuristic concepts out there to create a full-size holographic windshield for cars [18]. Although there is lots of interest from the industry for such technology, no such technology is commercially available, yet.

### 2.1.1 Augmented Reality in previous work

In previous research, AR has also been a topic of interest for applications within vehicles, and it has many potential applications. For example, AR has been used to study training scenarios [19] as well as in studies focusing on applying AR to explain the driving decisions made by an AV [20]. Other works focus on adding functionalities that cars do currently not support, such as conference

calls. In [21], a driving simulator was used to test the effects of using an HMD for AR video calling while driving a vehicle.

Various studies have focused on using AR to improve the ability of the driver to assess danger. In a study by Tran et al. [22], a proposal is presented for the usage of AR through a HUD to assist drivers in making left turns across oncoming traffic. In this given scenario, the driver must make many judgment calls to carry out manoeuvres in a safe manner. To validate their designs, they made use of a simulator that allowed for testing with and without aid from the AR system, so that the results could be compared (figure 3). Although they had a small number of participants (four that completed the study, three withdrew due to motion sickness), they still gained valuable insights into how to improve the interaction in the future.


**Figure 3: The simulator environment used in Tran et al. [22].**

In [23], it is explored how AR cues can be used to direct the attention of the driver to potential roadside hazards. In this study, participants were evaluated based on their response time in detecting hazards. For this study, a simulator was used as well. Based on their findings, they claim that AR cues did not distract the drivers or impair their ability to assess danger.

What's notable is that many of these studies, simulators have been created to test the proposed interactions with the users. In most of these cases, large screens were used with the aim of creating a more realistic effect.

## 2.2 Object Detection

Object detection is a computer vision technique in which semantic objects of a particular class (e.g., humans, cars, traffic signs) are detected within digital images and videos [24]. The obtained output of these techniques can be applied for anchoring augmentations on top of objects appearing in videos in real-time (figure 4). Therefore, this work is so relevant to be used within the simulator.

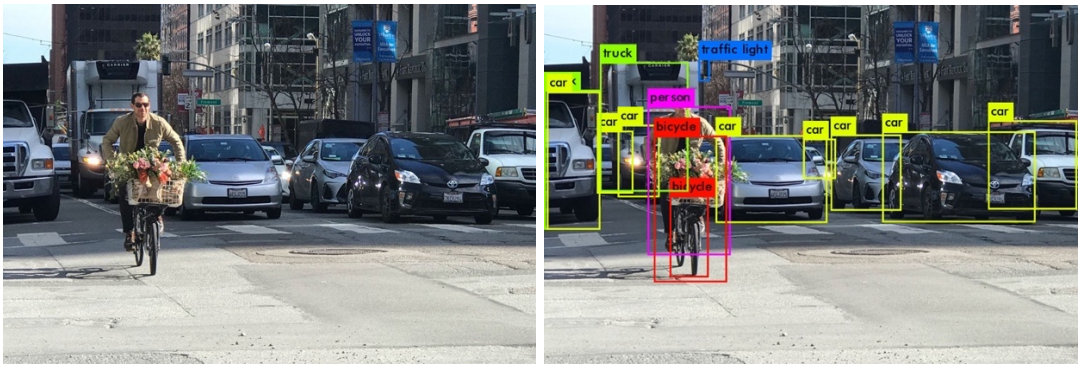



**Figure 4: the image on the left is the input-image given to Yolo. The image on the right shows the bounding boxes as well as the class-labels that were obtained through YOLO.**

In recent years, the capability of detecting objects in images has significantly increased thanks to new developments in Machine Learning (ML) by applying Deep Neural Networks (DNN's) [25]. As part of the advances in both ML and computing power, it is now possible to detect up to at least 80 classes in real-time (above 30fps) using hardware that is still considered "consumer-level" as has been shown in previous works [26][27].

## 2.3 Eye-tracking

Eye-tracking is the process of either measuring the point of gaze or the motion of an eye in relation to the head. An eye tracker is a device capable of measuring the eye positions and its movement. Modern eye trackers achieve this through video analysis. Modern trackers, project patterns of near-infrared light on the eyes of the person of which they aim to measure the gaze. Then, through image processing, the gaze points are calculated [28]. An example of such an eye-tracker is the wearable Tobii Pro 2 glasses (figure 5). Other eye trackers approach this by estimating purely on video [29]. Although these solutions don't require any external or wearable hardware, they tend to be less accurate.


**Figure 5: The Tobii Pro 2 glasses worn by a driver. Obtained from [28].**

The application of eye-tracking devices has also been used in research related to driving vehicles. Most of the identified research has focused on measuring eye gaze behaviours to detect if the driver is showing signs of fatigue or drowsiness [30][31]. Other studies have focused on using eye positions and pupil diameters to measure possible distractions in traffic [32]. Furthermore, other work has also focussed on devices present in the car itself, such as a navigation system [33] or while navigating a menu on the car's HMI [34].

# 3 Description of both systems

This chapter will describe the two systems developed as part of this thesis. First, there is the simulator environment that is used within the user research on AR for remote driving. Second, there is the Torino Demo system. This chapter will describe both systems from the perspective of the user. Technical details of its implementation will be discussed in the following chapters.

## 3.1 The Simulator Environment Overview

For the simulator, the main goal is to produce a setup that can be used to evaluate AR interactions. These AR interactions are within the context of improving safety and remote operation or monitoring of (semi-) autonomous vehicles. It does so by exposing the user to different videos of a vehicle driving through heavy traffic in different urban locations. For each video, a different type of 'AR assistance' can be applied to the video (or left out). The system can also deteriorate the quality of the video to simulate remote driving conditions in sub-optimal coverage areas. Users will watch these videos on a large screen in front of them, while wearing glasses that can track their eye-gaze. An overview of this system can be seen in figure 6.

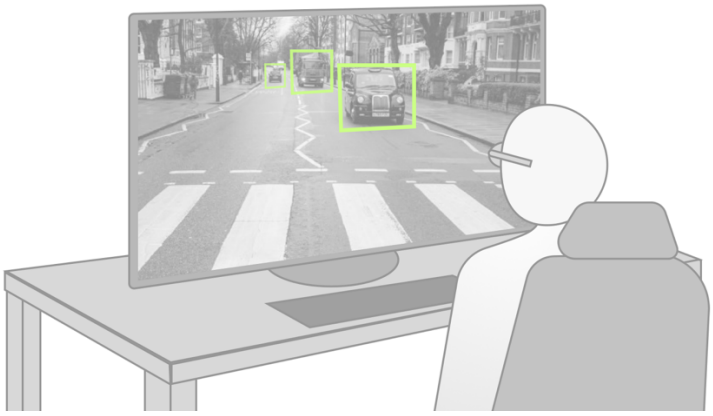


**Figure 6: The set-up for the simulator environment. Participants will be seated in front of a large display while wearing the Tobii Pro 2 Glasses.**

Users will be given the task to press the spacebar on the keyboard in front of them, in case they spot danger. For example, when a pedestrian jaywalks and the vehicle, they are monitoring would need to brake. As the user is observing each scene, and reporting for possible danger, the simulator works in the background to collect and store research-relevant data for later analysis automatically. The system observes where the participant looks on the monitor. Also, it notes what objects they are looking at and if they reported any dangerous scenario on a frame by frame basis (figure 7).

Although the aims of this work are focussed on safety, the set-up of the simulator was done in such a way that it could be adapted to other purposes with great flexibility. Furthermore, the selection and implementation of the used technology were done in such a way that the simulator could easily be translated into a real working system.

In its way, the simulator also aims to be a realistic reproduction of what could be achieved in the present day. This also implies that all augmentations are being applied in real-time to the video, meaning that videos can be swapped without the need for pre-processing in any way.

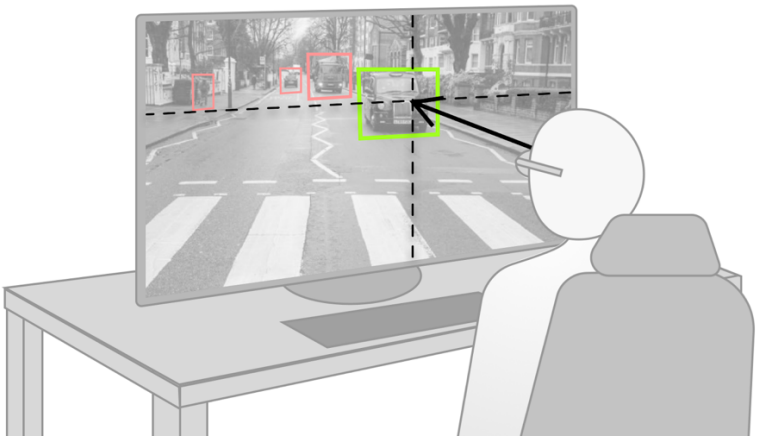


**Figure 7: The eye-gaze reading obtained from the Tobii Pro 2 glasses will be mapped to a coordinate on the monitor for and logged for every video frame. Additionally, it will be logged if the participant looked at an object.**

### 3.2 The Torino Demo Overview

The goal for this system is to simulate the experience of how, in the near future, ADAS could assist the drivers through unification of various ADAS systems. For the demo, participants are seating in the back seats of the vehicle. While a professional driver brings them along a set of scenarios and a guide in the front passenger-seat talks about each scenario, acting as a tour-guide (Figure 8). In this setup, the driver also serves as an actor that wears eye-tracking glasses and is supposed to act as an actual driver having access to these upcoming technologies. Unlike the simulator system, the goal is not to study the eye-gaze of the participant for the Torino Demo.

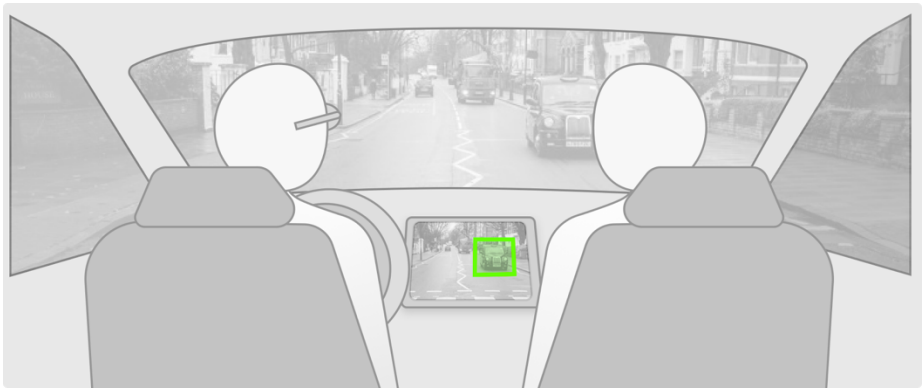


**Figure 8: The set-up for the Torino Demo. Participants will be seated on the rear-seats of the vehicle. Augmentations will be shown on the HMI of the vehicle.**

Just as for the simulator, the augmentations are shown on display. However, this time it's on the inside of the vehicle, on the HMI. A video stream captured by the dashcam will be shown alongside other elements of the car's HMI (which have been left out of Figure 8 for illustrational purposes).

However, note that the wearer of the glasses will be gazing at the objects outside of the car, rather than the objects on the screen (Figure 9).

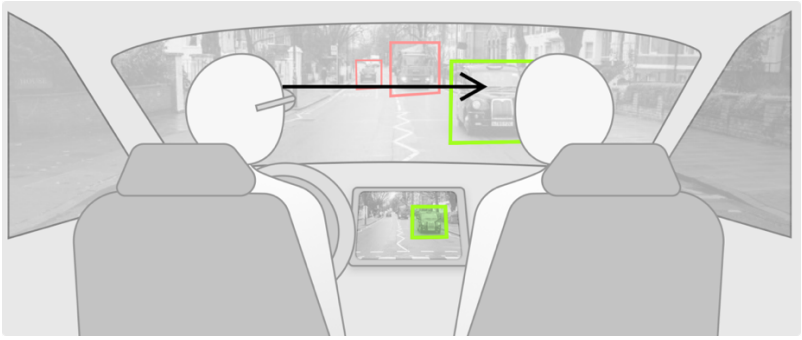


**Figure 9: The system will map the object being gazed at to the HMI inside the vehicle. This is where the augmentation will be displayed on the video feed obtained by the dashcam.**

## 3.3 Used Technologies

Both systems share a common set of technologies: object detection and eye-tracking. Furthermore, for both systems, openFrameworks [36] was used for bundling all pieces of the system together.

### 3.3.1 openFrameworks

openFrameworks (OF) is an open-source C++ toolkit distributed under MIT License. The framework is, according to its creators, meant to be used for experimentation and facilitating the creative process [36]. OF is a flexible framework that allows creatives and developers alike to leverage libraries such as OpenCV (Open Source Computer Vision Library [OpenCV Source]) while also integrating additional hardware such as cameras. Although the framework is extensive, in the sense of having its own “batteries included,” it is also a very extensible framework. There exist many add-ons for OF, and it has a relatively active community.

### 3.3.2 Object Detection

YOLO (You Only Look Once), is a Convolutional Neural Network (CNN) that is used for the object detection [26] in both systems. Essentially, what it does is take an image as an input, and as output provides the probability estimation wherein the particular image objects are present, as well as their dimensions (figure 10). This combination of location and size of an object, in an image, is also commonly referred to as a bounding box [44]. These bounding boxes can then be used as anchoring points for augmentations.

Although there are many ways in which object detection can be performed, YOLO offers a few significant benefits compared to other methods:

- The most significant benefit is speed: YOLO can perform real-time object detection - up to 45 frames per second - on an NVIDIA GeForce GTX 1080 Graphical Processing Unit (GPU);
- The network understands generalized object representations, so it also works on artwork. Within the context of the thesis, this meant that in theory, it was also able to differentiate between road signs (e.g., ones with cars or pedestrians on them);
- It is open-source and even has pre-trained weights, made available by the author and community contributors [27, 45].

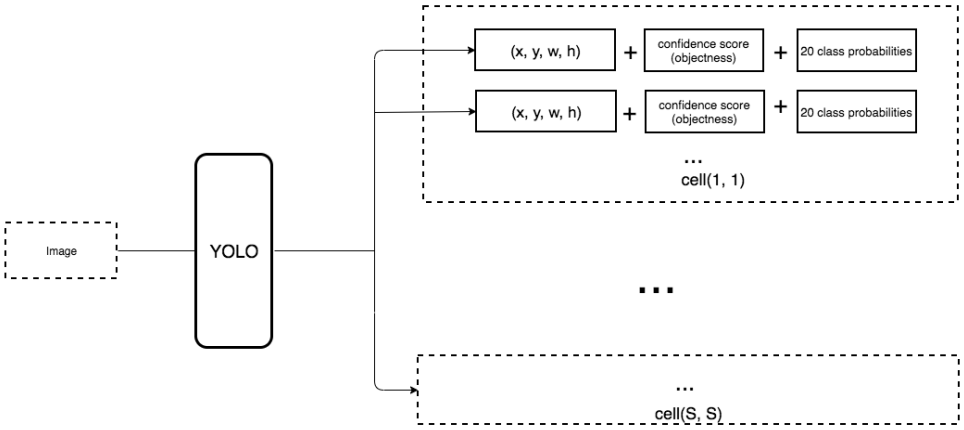



**Figure 10: illustrating the output obtained from the YOLO network. The network returns a vector containing all detected objects and provide their positions (width, height, x and y position) in the image as well as their respective label and probability. Figure was obtained from [53].**

### Implementing YOLO in openFrameworks

To integrate YOLO, an open-source implementation by Github user AlexeyAB was used as a starting point [43]. As with any neural network, one can use their training data, but in this case, the pre-trained models were sufficient. The model can detect a range of relevant objects that one finds in everyday traffic (e.g., people, cars, trucks, traffic signs, and traffic lights). Although one could use techniques such as transfer learning, in which the only classification layers are retrained, to improve the performance, this was deemed to be unnecessary.

Although the original repository by AlexeyAB provides some installation instructions, this process is not straightforward, and many pitfalls were discovered along the way. Therefore, the time has been invested in documenting more detailed installation instructions for others to use in the future. These are included in [46].

### 3.3.3 Eye-Tracking

The Tobii Pro Glasses 2 (from now on referred to as “the glasses”' for conciseness) is an eye-tracking device used in the thesis for eye tracking purposes [37]. These glasses consist of two parts:

1) The head unit, which captures the field of view of the wearer and measures the orientation of the eyes to determine the gaze location. This process is done with four cameras for each eye and measures at a rate of 50 to 100 Hz. The head unit weighs 45 grams, which makes them slightly heavier than an average pair of glasses.

2) The recording unit, which is a small box that is connected by a cable to the glasses. This box stores the calibration data of the wearer. It also acts as a streaming component for the video and data stream. This component weighs 312 g but, unlike the glasses, doesn't have to be worn as it can also be put on e.g., a table.

For the purposes of the project, this device offers a few essential benefits:

- It is a wearable eye tracker that looks and feels relatively similar to a regular pair of glasses;
- Offers eye gaze data about what the user is looking at in real time. This data is accessible through their device API and can be streamed in real-time;

- It is also equipped with a Full HD wide angle camera (H.264 1920 x 1080 pixels at 25fps), which when the glasses are worn are positioned just above the nose-bridge. This positioning provides an ideal perspective w.r.t to what the wearer is seeing. This camera feed can be streamed live over RTSP, which is a protocol for streaming data (such as video);
- As a device, the glasses are very easy to use for the user, as they don't need any training. Furthermore, the calibration process is quick and straightforward, which is a great benefit compared to other eye trackers.

**Implementing Eye-Tracking in openFrameworks**

Two main components had to be integrated within openFrameworks. First, there is the video stream from the camera on the head unit. This process goes relatively straightforward, as OpenCV [40] allows you to stream over RTSP (Real-Time Streaming Protocol), a video source through the VideoCapture class implementation [41].

For obtaining the eye gaze data, it is a bit more complicated. Luckily, there is an open-source controller for accessing eye-tracking data for the Tobii Glasses Pro 2 on Github, which implements their official API in an effortless way [37, 38]. However, this implementation is based on Python and does not directly interface with openFrameworks (which is based on C++). Therefore, the quickest way around this to stream the data from the controller over UDP (User Datagram Protocol) over a local port on the computer and read this port using openFrameworks' ofxUDPManager implemenation [39].

# 4 Technicalities of fusing Eye-Gaze and Object-Detection

This chapter aims to shed light on the challenges that were faced with regards to working with wearable eye-trackers in combination with object detection algorithms. Additionally, strategies that were applied to counter these challenges will be discussed as well. Although this thesis won't go into depth on all technical parts as much as this particular one, this topic is a core part of the technical efforts that have been worked on.

The first section deals with a problem that arose when dealing with small objects in a scene in combination with slight inaccuracies in the eye-tracking detection.

The second section describes two scenarios in which the eye gaze estimation had to be mapped to a different surface. This section consists of two parts. In the first part, it is about how the eye gaze estimation was mapped to a coordinate on a monitor. This was directly applied within the thesis to make data analysis possible through the simulator system. The second part describes how eye gaze readings were used to estimate which object was being looked at, from the perspective of a secondary camera. This was used within the Torino Demo to determine at which object(s) the driver was gazing.

## 4.1 Dealing with inaccurate gaze-readings

This section describes the implementation and strategy behind a measure for “correcting gaze readings” that were applied during the early stages of the work. This problem appeared while working on early exploration of what could be achieved with the available hardware. This work was done before defining the goals for both the remote driving study as for the Torino Demo.

The goal of this technical exploration was to try to combine YOLO object detection with the Tobii-glasses eye-tracking capabilities. In other words: could we create a demo set-up, that would demonstrate the integration of both components? And by doing so, what kind of expectations should we have of such a system with regards to its capabilities?

By having such a system, we could estimate the feasibility of further possibilities. Now, that it is possible to detect what object a user was looking at, it allowed for testing the reliability such a system would be for future use cases.

In its essence, this demo set-up did the following:

1) Obtain the current video-frame from the glasses and pass them into the YOLO object detection.
2) Obtain where the wearer of the glasses was gazing.
3) Test if the gaze-location of the wearer was inside of a bounding-box from a detected object.
4) Trigger a specific interaction (e.g., playing a sound) if the wearer was looking at a specific object.

Upon realisation of this prototype, it became clear that there were three problems:

- Detecting user-gazing at small objects accurately was troublesome. This was mainly due to small inaccuracies in the gaze-detection. This, combined this with a small bounding box could result in a false-negative scenario where the user's gaze was placed outside of the bounding box (figure 11 – left image);
- Bounding boxes are somewhat an awkward spatial-representation of real-world objects, as most of them - except for devices - do not exactly fit in the shape of a rectangle very well. This leads to false positives for larger objects in general;

- Smaller bounding boxes could be contained within other bigger bounding boxes: what if the user was looking at a mobile phone being held by someone? This, combined with the previously mentioned issues, resulted in false-positives for the larger objects and false-negatives for the smaller object (figure 11 – right image).


**Figure 11: Illustrating two issues with using identified through the first prototype that combined eye-tracking with YOLO Object-Detection.**

Therefore, the goal was set to implement a system that would compensate the bounding boxes somewhat to prevent the issues described above. The chosen approach was to make use of a simplified set of "Steering Behaviours," as described by Reynolds [50].

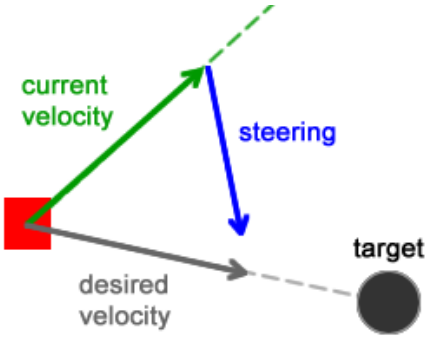



**Figure 12: Figure illustrating the force-vectors that are applied to Reynolds' vehicles. Figure obtained from [50].**

Within this model, movement is modelled as a "desire" toward an object, which results in a force being applied to the agent, which in turn causes the agent to move towards its desire(s) (figure 12). This steering-vector can be used within the system to make assumptive corrections w.r.t eye-gaze readings. The benefit of this model is that multiple objects can emit a desire to the vehicle at once, and the movement is a result of the force applied by the sum of the desires emitted by every object. Furthermore, the system is time-based which means that corrections made by the desires (the bounding boxes), happen over time. In practice, what this means is that when the user keeps gazing at an object, the correction effects get more expressive over time. Meaning, that if a user keeps waiting - in anticipation of a response from the system - the steering behaviours will correct the estimated eye-gaze position (figure 13).

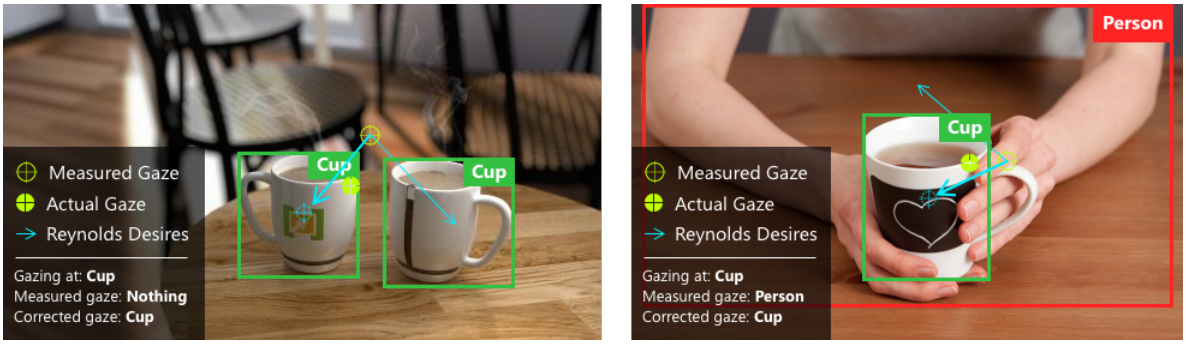



**Figure 13: Illustration of how coupling the eye-gaze coordinate to a Reynolds' vehicle fixes previous discussed issues with detecting eye-gazes at small objects.**

To implement this, the following changes were made to the demo:

- The eye-gaze estimation, used to determine where the user was actually gazing, was decoupled from the measured eye-gaze provided by the glasses. The measured gaze, provided a steering behaviour for the estimated eye-gaze by applying a force to the estimated gaze position;
- Bounding boxes also applied a force to the estimated eye-gaze. The smaller the bounding box, the bigger the force they applied. Bounding boxes only provided their force once the estimated eye gaze was in near proximity to them.

An example implementation can be found on the GitHub at [47], the code itself is reusable for openFrameworks based projects and comes with an example so that it's effects can be tried out. In the end, for the thesis, these strategies were not applied in the final deliverables. The reason for this is that over time the requirements for detecting "relatively small objects" and "small objects within bigger objects" were no longer relevant.

### 4.2 Mapping eye-gaze to a different surface

In this section, two scenarios will be discussed in which the eye-gaze readings are mapped to a different image than the one being provided by the glasses. The first case is about a scenario that happened during the user research, and the second is about a challenge that arose during the set-up for the Torino Demo.

For the user research using the simulator environment, participants will be watching a video being played on a computer screen while wearing the Tobii glasses. The main data that the system needs to be able to derive here is which objects in the video did a user (not-) see, and where exactly was their gaze focused on a given point in time. In order to allows for this, the system needs an understanding of what objects are currently in the video, as well as which position the gaze of the participant is fixed on. The main challenge here is mapping the gaze position (*real world*) to a position on the screen (*virtual*).

For the Torino Demo, the driver of the vehicle will be wearing the glasses while encountering various sets of events. Some of these require understanding of what is happening outside of the vehicle (*real world*) in combination with understanding where the driver's gaze is located (*real world*, but *different perspective*).

Both cases have two things in common:

1) Essentially, there are two video inputs for each scenario. In the case where the participant is looking at a video on screen, one input source is the video being displayed, while the other

input comes from the video-stream from the camera on Tobii Glasses. However, for the Torino-demo the glasses will be worn by the driver of the vehicle while the second video source is provided by a dashcam installed on the vehicle.

2) They share the goal of mapping a gaze position obtained from the glasses, to another surface. In the case of the user study, the goal is to map the gaze from the glasses, to a position on the monitor. While for the Torino-demo, the goal is to map the gaze position of the glasses, to a dashcam capturing objects in front of the vehicle.

The first part of this section discusses the issue of mapping an eye-gaze position to a coordinate on a monitor screen. The second section describes a simple strategy that was applied to map “roughly” the eye gaze position to a different camera. Additionally, unapplied measures that could have been implemented for better performance, are also mentioned.

### 4.2.1 Gaze position to monitor-coordinate mapping

For the user research using the simulator system, participants will be seated in front of a large screen while their gaze position on the screen will be measured. Additionally, it will track what object is being gazed at (e.g., a pedestrian or a vehicle).

One may wonder, why the glasses were used at all for this specific set-up. Especially, given that Tobii also has a non-wearable product in its portfolio for detecting the gaze position on a computer display. Although this option was considered, it was deemed to be better to use the glasses. The reason why this device configuration did not suffice is that it doesn't work as well for a large display that would necessarily cover most of the participant's field of view. This was a design-driven decision given the goal was to emulate the conditions of being in a real-car, and therefore having a large field of view was deemed more valuable. Furthermore, being able to accurately measure the eye gaze location on the screen also reduces data analysis efforts. The coordinates can be stored in a document, automating a part of the data gathering.

Now, the question remains how the eye gaze location can be accurately mapped to a different surface, such as a computer screen. This location can be approximated by mapping a 2-D vector (the eye-gaze reading) contained within a plane (the video-frame from the glasses), with a non-fixed orientation in 3D-space, to a plane with a fixed orientation in 3D-space (the monitor). The exact position of the monitor depends on the participant, factors that influence this are: their height, distance to the monitor, and such. An overview can be seen in figure 14.

To make this mapping successful, it is necessary to know where the eye gaze intersects with the computer screen, which can be obtained by knowing the orientation and position of the camera in the glasses w.r.t to the monitor. There are multiple ways to achieve this, but it was decided to do this by using an ArUco marker [48], displayed on the screen, as a reference point. The main reasons to select the ArUco marker for this effort are:

- The detection of ArUco markers did not require any additional hardware and was therefore both inexpensive and fast to implement;
- Markers can be detected extremely fast allowing it to be done in real-time. This has the benefit that in case the user would move during the experiment, the results would still be accurate;
- The original library is written in C++ and is supported through the community version of OpenCV, a library that was already used within the project;
- It can easily be scaled to multiple displays, as each ArUco marker also has an unique identifier. Therefore, the same solution could work for a set-up with multiple monitors as well;

- Using YOLO to detect the screen coordinates wouldn't work here: the possible tilt of the screen cannot be considered. By using an ArUco marker, this tilt can be included as well.

Before you can make use the ArUco library, it is necessary to calibrate the camera that you're going to use. A camera can be calibrated through the ArUco library as well by taking pictures in various poses w.r.t to a ChArUco board (figure 15). Which is essentially a chessboard with ArUco markers contained in the white squares. During this camera calibration process, the intrinsic parameters and distortion coefficients are obtained [49]. This process only needs to be done once if the camera optic is not modified (which is the case for the glasses, as its camera does not change its optic). Once the camera is calibrated, it is possible to detect markers.

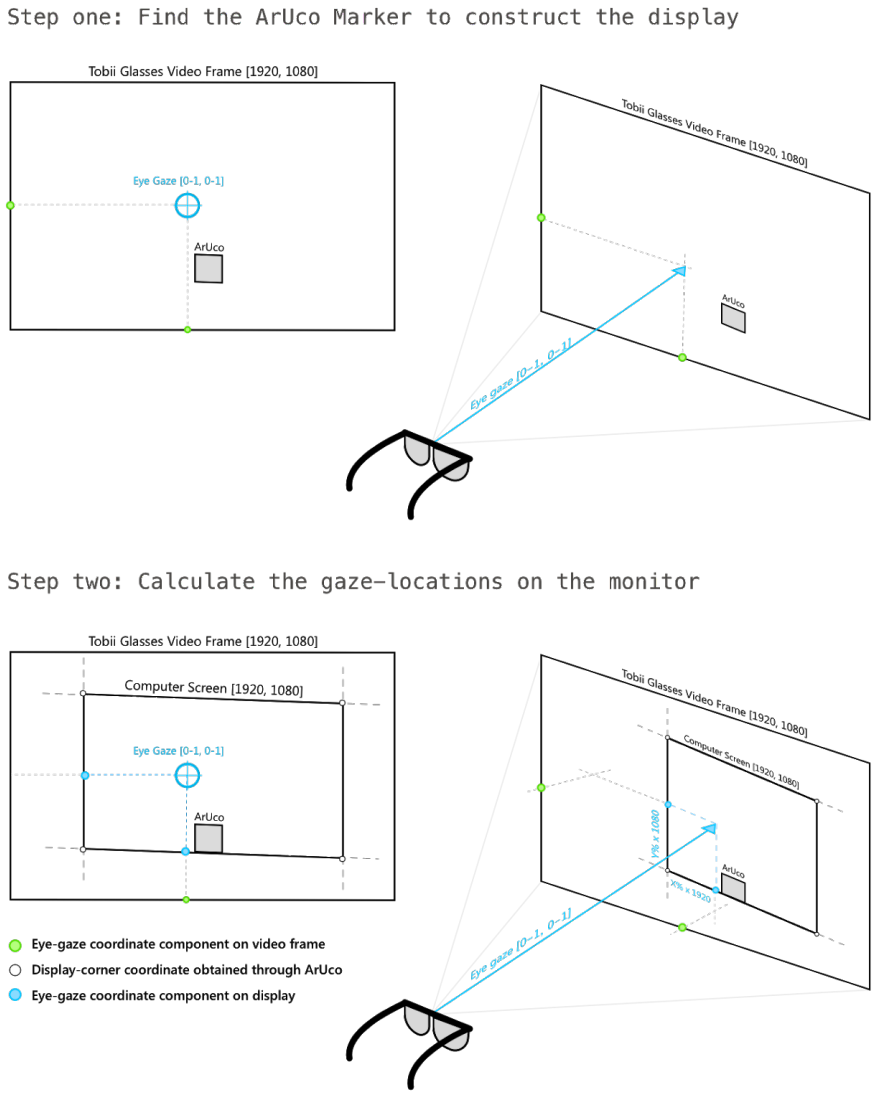



**Figure 14: Illustrating the process of mapping an eye-gaze reading on the video-frame to a coordinate on the monitor displaying the video in the simulator environment.**

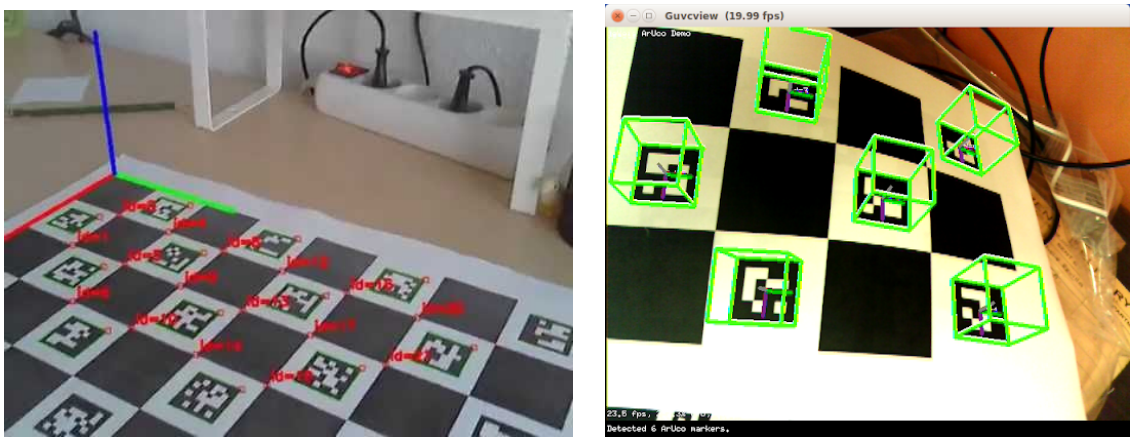



**Figure 15: A printed CharUcoBoard that can be used to calibrate the ArUco classifier.**

The following pipeline was implemented to determine if the participants gaze was within the bounds of the monitor, and at which location:

1) First, the video frame obtained from the glasses is converted to an OpenCv Matrix. Then, it's passed on to the detectMarker function in the ArUco library. Which returns a vector container in which each element contains the properties of the marker. Note, this also implied that the system updates the position of the monitor in the video-frame for every frame.
2) By finding the ArUco marker in the videoframe, four coordinates are obtained: one for each corner of the marker. Based on these corners, two orthogonal vectors can be obtained: vector-A (bottom left and bottom right corners) and vector-B (bottom left and top left corners). These vectors have a magnitude equal to the relative size of the marker w.r.t to the video-frame.
3) Then, by multiplying the magnitude with a fixed scalar, these two vectors can describe the size, location and orientation of the monitor. The size of this scalar depends on the relative size of the marker w.r.t to the monitor.
4) Then, vector-A is moved to the left corner of the monitor by a fixed factor, while vector-B is moved to the centre of the marker (in between the bottom left and bottom right corners).
5) These two line-vectors can now be used as a relative-axis system to determine the eye-gaze location on the screen. By finding a point that is both closest to the eye-gaze coordinates within the plane describing the monitor and containing the line-vectors, the X and Y coordinates can be obtained. Note, this is an approximation that works due to the nature of the experiment's set up, as it can be assumed that videoframe and the plane describing the monitor are (nearly) parallel to each other.
6) Finally, in case the closest point on the line-vector, to the eye-gaze location is either the start or end point of either vector, it can be assumed that the location of the eye gaze is not within the monitor.

As for the purpose of research within the thesis, the implementation was left at this level of technical fidelity. There are a few things that can be taken in consideration for possible future work to improve the current pipeline:

- Currently, the coordinates on the screen are estimated based on finding the point on the line vectors that is closest to the location of the eye-gaze. However, this is only accurate

when the wearer of the glasses is positioned relatively parallel to the screen. It would be more accurate to take a possible perspective-correction w.r.t to the screen in consideration as well. By doing so, the pipeline should refer to the orthogonal line intersection with the line-vector, rather than the point on the line-vector that is closest to the eye-gaze position.

- For this pipeline to work, the ArUco marker must always remain visible. Which unfortunately, also results in the participant seeing the marker as well, at all times. A more elegant solution would be to hide the marker somehow or choose for a different system altogether that is less visible to the naked eye.
- In theory, this system should work for an arbitrary number of surfaces, as long as you don't run out of ArUco marker-IDs, you're good to go. However, the current code-base only works for one specific ArUco marker as this use case was not taken into further consideration. For future work, this presented pipeline might be an interesting starting point.

### 4.2.2 Mapping a gaze position to different camera

During a specific part of the Torino Demo, the challenge was to demonstrate that it was possible to register that the driver had seen a pedestrian crossing the street on the HMI of the vehicle. Now, the Tobii glasses provide their own video-stream. However, due to specific reasons, it was decided that the video stream of the dashcam was to be displayed on the HMI.

Phrasing it differently, the glasses had to be used to detect if the pedestrian had been seen, but the feedback had to be shown on a different video, with a different perspective than the glasses. What makes this challenging, is that the video-stream that is used for the detection if the pedestrian had been seen, has a different outlook on the scene than the dashcam does. Furthermore, simply testing if a "person" had been seen, wouldn't suffice as looking at once hands would also trigger this.

Do note that the approach described in this section is intended for a controlled environment in which there is an object - such a pedestrian or a vehicle - that should be highlighted on the onboard HMI. Unlike the method described in the previous section, the goal is not to obtain a precise location. Rather, the intention is to derive which object outside of the vehicle is being gazed at. As for the purpose of the Torino Demo, this is deemed enough. Coming at the cost of having a lower accuracy and precision; the method does not require any special referential markers. This was considered a major benefit given that the implementation should be visibly hidden from participants.

In short, the method intends to isolate detected objects outside of the vehicle - specifically pedestrians - and determine if the gaze is focussed on them. Should the gaze be focussed on such an object, it should be highlighted on the HMI:

1) The first step runs the object detection on both video frames - from both the dashcam and the glasses - and stores the results in a separate vector.
2) The next step is to filter-out any objects that are not of the right class-label. All detected objects that are not pedestrians are filtered from both vectors.
3) For the vector that stores the result coming from the glasses, it is essential to remove false positives. The most common false positives in this set-up are the hands of the driver on the steering wheel, as these are classified with a "person" label as well. Therefore, the system needs to be capable of discriminating between hands inside of the vehicle and the pedestrians outside. The easiest way to remove those is to only consider detections with a human label that have their bottom coordinate above a certain height. Another factor to take into consideration is the area covered by the bounding box. The driver's hands tend to produce a way bigger bounding box than pedestrians do (due to relative distance).

4) Finally, in case the system needs to account for multiple pedestrians, it may need to disambiguate between them. An easy implementation to solve this is to sort the bounding boxes in both vectors (e.g., by top-left coordinate of the bounding box) so that disambiguation becomes possible (figure 17).

Now, this system makes distinctions based on very elementary properties. It works for a controlled environment, but it won't suffice for a system in the real world. For further accuracy in discriminating between pedestrians (and false alarms like hands), the system could also take factors such as colour-histograms into consideration. Another aspect would be to take overall luminance around the eye gaze location into account, as it's likely to be brighter outside of the vehicle (during the daytimes). This could also help to discriminate between looking outside and people inside of the vehicle. These further optimizations were not implemented, as this was not considered relevant for the Torino Demo.

**Figure 16: Illustrating the set-up for the Torino Demo where two camera's are involved. The top image shows the perspective from the Tobii Pro 2 Glasses. The bottom image shows the perspective from the dashcam onboard of the vehicle.**

**Figure 17: An image captured during an early iteration of the work. On the left image, the perspective from the Tobii Pro 2 Glasses can be seen. The small blue box is the current gaze-point as obtained through the glasses. The image on the right, is the video-frame from the dashcam. The object the user is gazing at on the left video frame, is augmented on the right frame by the technique discussed in this section.**

# 5 Implementation of the simulator environment

This chapter describes the main dependencies used within the simulator environment. The first section covers technical details, such as the used hardware and software, allowing others to recreate or repurpose the set-up. If this is not the intent or interest of the reader, this part can then be skipped. The second part focuses on the implementation of the simulator itself.

**Figure 18: An overview of the simulator environment displaying a video that used within the user-study.**

## 5.1 Dependencies

This section covers the hardware and software that were used for creating the simulator environment that was later used during the user study.

### 5.1.1 Hardware

The application was running on an Alienware 17 R5 laptop, which at the time of writing is considered a high-end computer, running the Windows 10 operating system equipped with 16GB of installed RAM, an Intel Core i7 8-core CPU, and an NVIDIA GTX 1080 GPU. This high-end hardware is required for ensuring the application can do all the processing necessary in real-time so that the simulator can render the scene at 30 frames per second minimum.

To measure the eye-gaze of the participants, a Tobii Pro Glasses 2 [37] was used. This mode provides an 82° horizontal and 52° vertical visual angle, which is large enough for the given set-up. Furthermore, the eye-gaze can be sampled either in 50 or 100Hz, which means that for every video-frame that is shown to the participant, a new sample can be taken (given that the simulator environment will run at 30 frames per second).

For displaying the application to the user, an external display was used; a 40-inch Samsung UE40D8005 LED-TV.

### 5.1.2 Software

The simulator was developed in C++ using openFrameworks (OF) version 0.10.1 as its main framework. A few modifications have been made to OF specifically:

1) First, OpenCV version 3.4.6 was used instead of the OpenCV version that is shipped with OF out of the box. The primary motivation to use this version instead, was that it allowed for the detection of the ArUco markers. This was achieved by compiling the official 3.4.6 release together with the community add-ons, which includes the ArUco functionalities.
2) Second, in order to allow OF to interface with this version of OpenCV, a wrapper called ofxCV was used, which is an OF community add-on [51]. In order to achieve real-time object detection, Yolo v3 was used [26]. For this set-up, the pertained weights provided by the original creator of Yolo J. Redmon, were used for running the network classifications [Darknet weights]. To speed up the detections of YOLO, the GPU-accelerating library CUDA 10.0 was installed using cuDNN 7.2.
3) Third, the obtained gaze readings from the glasses were sent over UDP on a local address and read into main OF-application using ofxNetwork library, a networking library from openFrameworks.

## 5.2 Implementation overview

This section provides an overview of the workings of the simulator. The most critical components of the system are presented in a chronological order in which they are programmatically executed as well.

### 5.2.1 Scene and augmentation configuration

In essence, the simulator environment, plays a set of videos in a defined sequence. Before the simulator begins, it is possible to configure in which order the video scenes should be displayed. They will be displayed one after the other. However, in between scenes, the system temporarily pauses itself, which allows for displaying a tutorial screen of what is to come when the play function is resumed. Additionally, for each video scene, multiple factors can be preconfigured:

- The video quality can be reduced if desired. This can be used to simulate the (harmful) effects of having sub-optimal data streaming conditions;
- A predefined augmentation can be added to the scene. This can be used for flexibly testing and comparing the added information augmentations across the scenes.

This was set up in such a way that it can be used to simulate augmentations based on information from camera-based object detection. However, a custom video can be used just as well if the augmentation effect is added through video editing software manually. This would be most interesting in case the augmentation-layer has requirements that cannot be met with currently available technology yet.

**Figure 19: Augmentations applied to pedestrians can be seen in this image.**

### 5.2.2 Object Detection

During the program loop, the simulator will grab the most recent frame from the video that is being played. It will then pass it through the object detection pipeline to obtain the bounding boxes of the objects recognized within the scene. The neural network better known as 'YOLO' is used for this object detection. Note that if the quality video is set to be reduced, the object detection will still make use of the full-quality version.

### 5.2.3 Drawing the scene

During the rendering process, the simulator draws the scene in various layers. From the bottom up, this is how the scene is rendered:

1) First, the current video frame will be drawn in the level of quality that is predefined.
2) Should a scene be playing that is configured to make use of any form augmentation, then this layer is drawn next. The augmentations are drawn directly on top of the latest video frame.
3) An ArUco marker is drawn in the bottom-centre of the screen for obtaining eye-gaze data.

### 5.2.4 Measuring the eye-gaze

When the video scene is playing, the simulator will make use of the ArUco marker in combination with the Tobii Pro 2 glasses to determine where the research-participant is gazing on the screen. A more detailed description of this process can be found in section 4.2.1.

### 5.2.5 Logging data

As the participant is observing the scene, the simulator can store relevant metrics that allow for data analysis later. For example, of the current video scene that is being played, it can be logged where the participant is gazing on the screen. If the gaze was focussed at an object (detected by YOLO), then this can be registered as well.

Naturally, for every frame, it can also log what scene was currently playing, at which video-quality, and which augmentation effect was added to it. In essence, this allows one to reconstruct the scene, as well as indicate where the user was gazing at that given frame.

# 6 Experiment methodology

The purpose of this chapter is to give an overview of the method used for the experiment to collect data for evaluating the AR interactions using the simulator system.

First, the research questions and related hypotheses are presented. The second section covers the used data collection techniques. The third and last section includes the experimental design of the study.

## 6.1 Research questions and hypothesis

The goal of this experiment, is to find an answer to the following questions:

1. Can the use of AR compensate for situations in which the quality of the video is dramatically lowered?
2. Can the implementation of AR be used to guide the user's attention to the most critical situations in traffic?

Based on these questions, the following hypotheses are developed for the experiment:

H0. In case the video quality is lowered, participants will be more capable of detection danger when AR is applied to augment pedestrians and vehicles.

H1. In case the video quality is lowered, participants will need to scan the scene less in case AR is applied.

Participants will each be shown four videos fragments, amongst which there are two variables that are varied amongst them:

1. The quality-level of the video that is shown to the participants (100%, or 10%, where 100% implies shown in high definition - HD).
2. The type of AR that is used to augment pedestrians and vehicles in the video (more on this in section 6.3.2).

## 6.2 Data Collection

In total, 16 participants took part in the experiment. All participants were recruited from the same country of origin (China). The benefit of recruiting participants from a similar background and comparable driving experiences is that it would eliminate possible biases introduced due to cultural differences w.r.t traffic experiences [52]. All participants were in possession of a Chinese driver's license. There was no direct preference to recruit Chinese participants specifically. However, an opportunity presented to recruit participants from a relatively similar background through a Chinese student community. Given that past experiences and cultural background influence danger-estimation in traffic, this was seen as a benefit rather than a disadvantage for the research. Cinema vouchers were given to all the participants, as a 'thank you' for their contribution.

## 6.3 Experimental Design

This section discusses the testing set up, the augmentations that were tested during the study, the overall testing procedure, and finally, the data that has been collected.

### 6.3.1 Test Environment

The experiment was divided into four scenarios. In each scenario, the participants were shown a different video fragment of a vehicle driving through the city of Boston, USA [54]. The order in which these videos were shown was changed per participant. During the third and fourth sections, augmentations were also applied to the video. Each video had a length of roughly two and a half minutes. In total, the experiment lasted approximately 15 to 20 minutes.


**Figure 20: The overview of the structure of the user-study.**

The videos were recorded with the camera positioned on the hood of the vehicle, which almost provides a first-person perspective. This allows for a good overview of the events happening in traffic. Additionally, this video material was selected thanks to the high quality of the video, the passive and errorless driving style of the driver, and the contrasting disobedience of traffic-laws by the other traffic participants.

The task that was given to each of the participants was to observe the traffic situation presented in the videos and mark possible events that they would classify as a dangerous event. A dangerous event was defined as an event to which they, as a driver, would need to respond (for example, a pedestrian jaywalking in front of your vehicle that makes it necessary for you to reduce your speed to prevent a collision).

Participants were able to mark a dangerous event by pressing the space bar on the keyboard in front of them at the moment they identified one. Before the start of the experiment, participants were familiarised with their tasks to be performed during the set-up phase of the experiment. In-between different videos, a tutorial screen was displayed that reminded participants of their task, as well as what type of augmentation would be shown. This message was accompanied by a preview frame of a scene illustration of what the augmentations would look like in the next part.

### 6.3.2 Augmentation Effects

Although the order in which each of the video-fragments was shown to the participant was shuffled, the sequence which certain augmentation effects occurred remained consistent.

**Test 1 - High Quality Video - No Augmentations.**

The purpose of this scenario was to register the eye-gaze of the participant, without any possible augmentation effect interfering with their natural observation behaviour. Additionally, it also provided an accessible and relatable entrance to the experiment and its set up. The cumulative results of this testing scenario would later be used during the analysis phase as a base-case to compare the following scenarios.

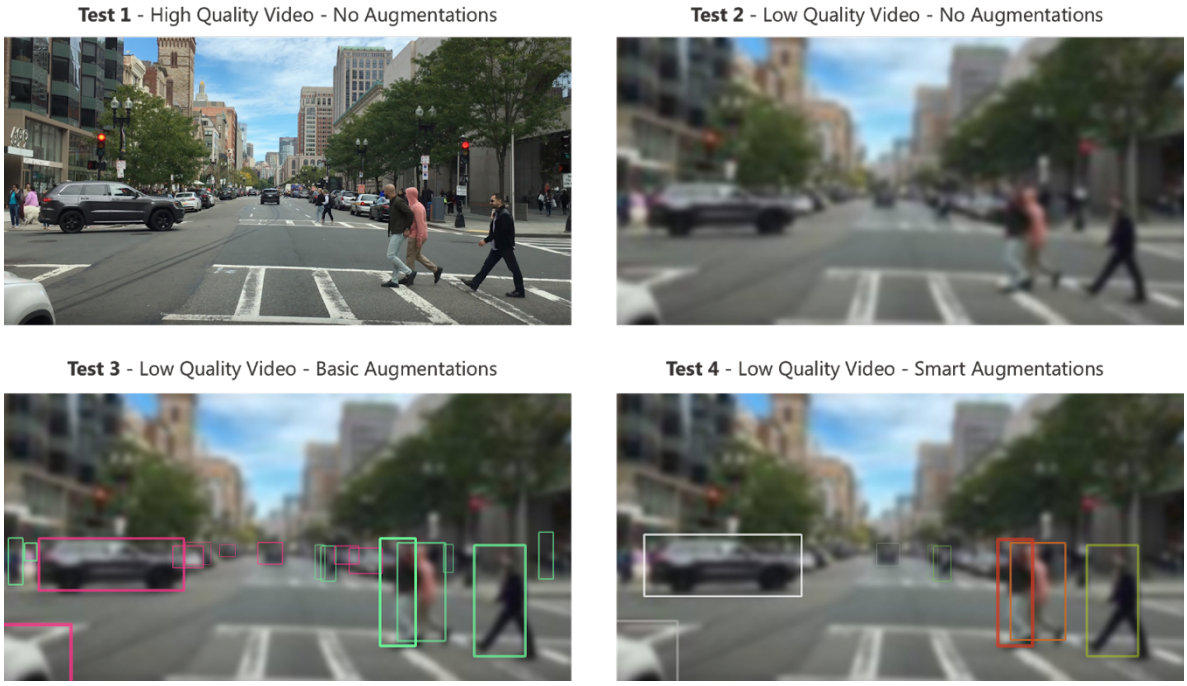



**Figure 21: Illustrating the four different visual-effects used in four sections of the user-study**

### Test 2 - Low Quality Video - No Augmentations.

For this scenario, the quality of the video was reduced — by 90% compared to the first video — to create a 'blurry-effect' in which details are eroded from the scene. The aim was to measure how participants observe a situation in case vision conditions is sub-optimal. Also, it would provide insight if participants would be able to recognise danger just as well as they did in the previous part.

### Test 3 - Low Quality Video - Basic Augmentations.

In the third scenario, augmentations will be shown for the first time in the user test. Both cars and pedestrians will be given a distinctive and bright border to make them stand out from the scene (no matter where they appear on the screen). Vehicles are given a violet border, while pedestrians are given a mint-green border. These colours were selected for both their high contrasting values with the video and the other type of border colour. Like the previous test, video is shown in a lower quality.

### Test 4 - Low Quality Video - Smart Augmentations.

In the last scenario, another type of augmentation is applied to the video. Unlike the previous round, augmentations are only shown depending on specific conditions. First of all, augmentations are shown in case the object (be it a pedestrian or vehicle), is in a direct line in front of the vehicle. Vehicles are given a white border with a lightly reduced opacity level. The opacity level and border thickness increase if the vehicle is in the same lane and gets closer to the vehicle. The border thickness also increases if the vehicle is approaching the driving lane from the side without slowing down (e.g., at a crossing), as can be seen in the example with the vehicle on the left. Pedestrians are given a specific colour, depending on how close they are to the car, and how close they are to the centre of the vehicle. Pedestrians that are close to the centre of the vehicle and are in close proximity

are given a red border. In case they are further to the side, the border will fade from orange to green. In case they are completely on the side of the road, no border is shown.


**Figure 22: This figure illustrates the location of the "danger-zone". Pedestrians that walked in this area, were given a red border colour (instead of green) as they moved closer the vehicle.**

### 6.3.3 Test Procedure

Before the start of the experiment, participants signed a consent form in which the experiment was explained. Afterward, an explanation was given about the procedure of the research. This was also the moment that Tobii Pro 2 glasses were given to the participants, and they were asked to wear them.

This was followed up with the calibration procedure for the glasses. During this procedure, participants are asked to look at a calibration graphic for a few seconds. Once the glasses were calibrated to the participant's sight capacity, the experiment started. After the experiment, an unstructured interview was held to inquire about the participants experience with the augmentations in general.

### 6.3.4 Data Collection

For data analysis purposes, metrics were automatically logged into a CSV file during the experiment by the simulator. For each test, the following metrics were saved for every individual frame of each video shown: the participant ID, the screen dimensions, the video that is being played, the video-frame that is visible at the time of logging, the quality of the video frame, if the participant saw an object, if the participant gazed at a person if the participant gazed at a vehicle.

### 6.3.5 Trial Run

Prior to the start of the experiments, a trial run was organized in which 8 participants participated. The goal of this trial run was twofold. First of all, it was done to confirm that the simulator worked as expected for when the real experiment would take place. The second reason was done out of curiosity what would happen if the video quality would not be reduced during the experiment. As part of this trial run, four of these trial-participants didn't experience the second,

third and fourth segments with a reduced video quality. Instead, they were shown all segments in a high video quality (see figure 23 for an overview).

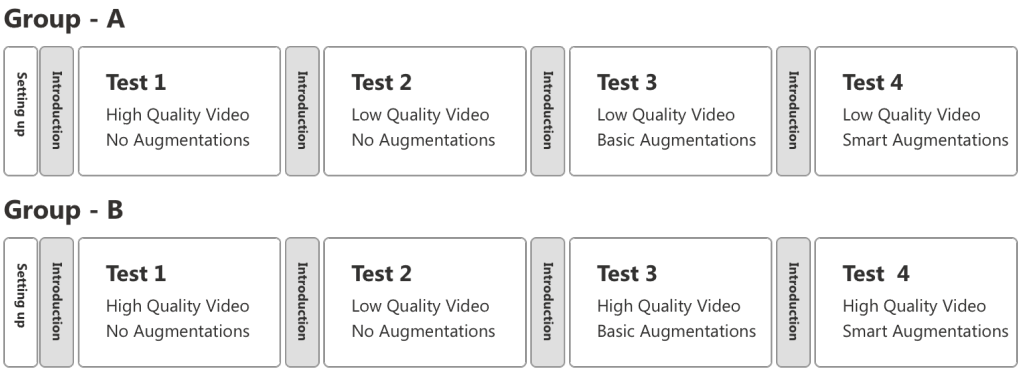



**Figure 23: An overview of the trial run's testing procedure in which 8 participants participated.**

However, upon analysis of the data, it was found that the differences in eye-gaze behaviour amongst Group-A was more significant compared to Group-B. Therefore, it was decided to continue the experiment as planned and use all available participants to test the effects of the augmentations on a video with reduced video quality.

# 7 Results and Analysis

This chapter discusses the analysis of the data obtained through the user study. The first section discusses how the validity of the collected data was assessed, and how decisions were made with regards to what data could be used for what purposes. The following section discusses the analysis and comparison of eye gaze behavior for each of the four tests that have been conducted during the user study. The third section will discuss the analysis and findings from the events marked as 'dangerous' by participants and compare these across the tests. Finally, an overall discussion of the findings can be found at the end of this chapter.

## 7.1 Assessing the validity of the data

One factor that came to light while visualising the raw data, through using MATLAB's plotting tools, was that for six of the participants the eye gaze measurements were likely recorded incorrectly. The cause is most likely their own glasses that some participants requested to wear underneath the Tobii Pro 2 Glasses. Given that this affects the eye-gaze readings, it was decided to not make use of this data for the comparison of eye gazing behaviour across the tests. The data was still considered useful for the comparison of dangerous events across use cases and was included in that part of the analysis. An example of valid data can be seen in figure 24. An example of faulty data can be seen in figure 25.

## 7.2 Comparing eye-gaze behavior

The first step taken to analyse the data was to make use of a three-dimensional histogram that expressed the eye gaze behaviour across all of the participants, for each type of augmentation. To achieve this, all the eye gaze data - that were measured during the same type of augmentation - were concatenated into one data set. As for visualisation, it was chosen to use a bin size of 120x120 pixels. By using this configuration, 16 bins fit in the x-dimension, while 9 fit in the y-direction (based on the screen resolution 1920:1080). This effectively divides the screen in squares of which the volume indicates the quantity of gaze points that are recorded within these coordinates. In figures 26-29, an overview can be seen of the obtained histograms.

Across each of them, the area with the most measurements - the highest bins - is roughly just below the centre of the screen. This is also rather explainable given that this is the height of the horizon, and that while driving in traffic, most things you should pay attention to happen in front of you (e.g., other vehicles in the same lane or a crossing pedestrian). Based on these histograms, a few additional observations can be made when comparing it to the base condition (1), which had no augmentations and displayed the scene in high video quality:

1) For (2) and (3), noticing the wider spread of gaze measurements, it is apparent that eye gazes were not as focussed as much as they were on the centre during (1).
2) For (4) there is more focus around the centre of the screen. The spread of gaze points is less wide then it is for (1), meaning the participants were mostly gazing at the centre.

One possible explanation, for why during (4) the participants were more focussed with gazing around the centre of the screen is that this was the area from where the augmentations could be observed. Unlike for (3), where augmentations were also applied, the augmentations were only visible during (4) once the object was near the centre of the screen as well.

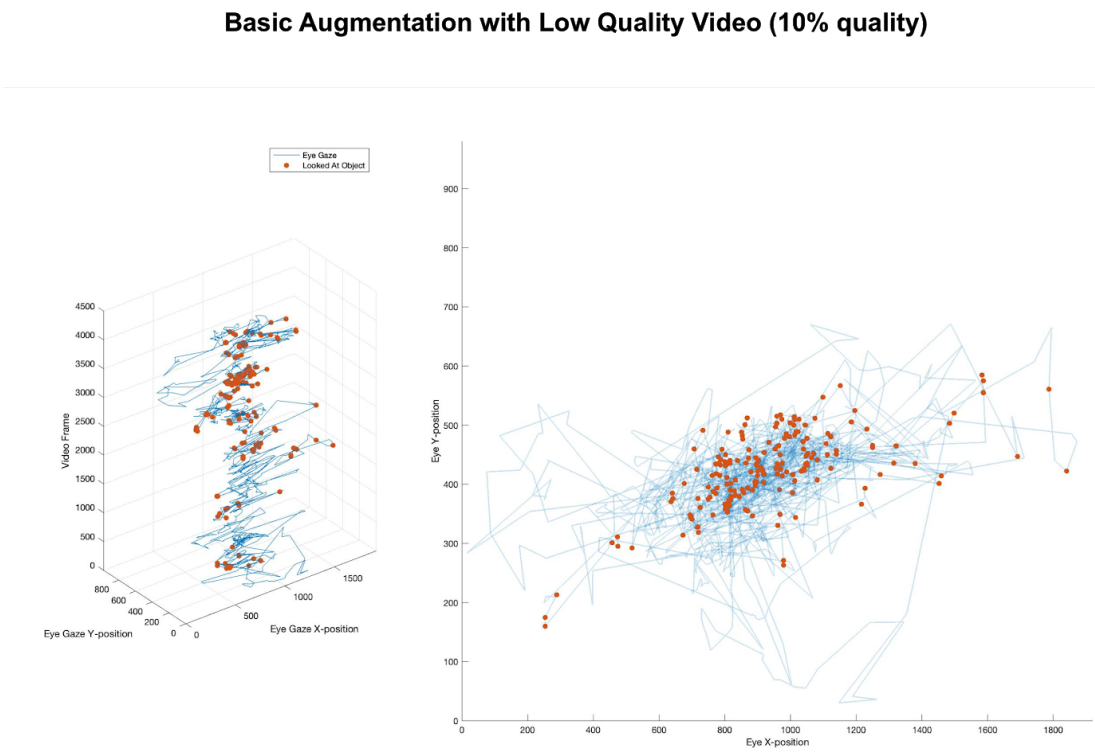



**Figure 24: Raw data obtained during the experiment. The blue lines illustrate the eye-gaze path. The red dots indicate frames in which the participant gazed at an object.**

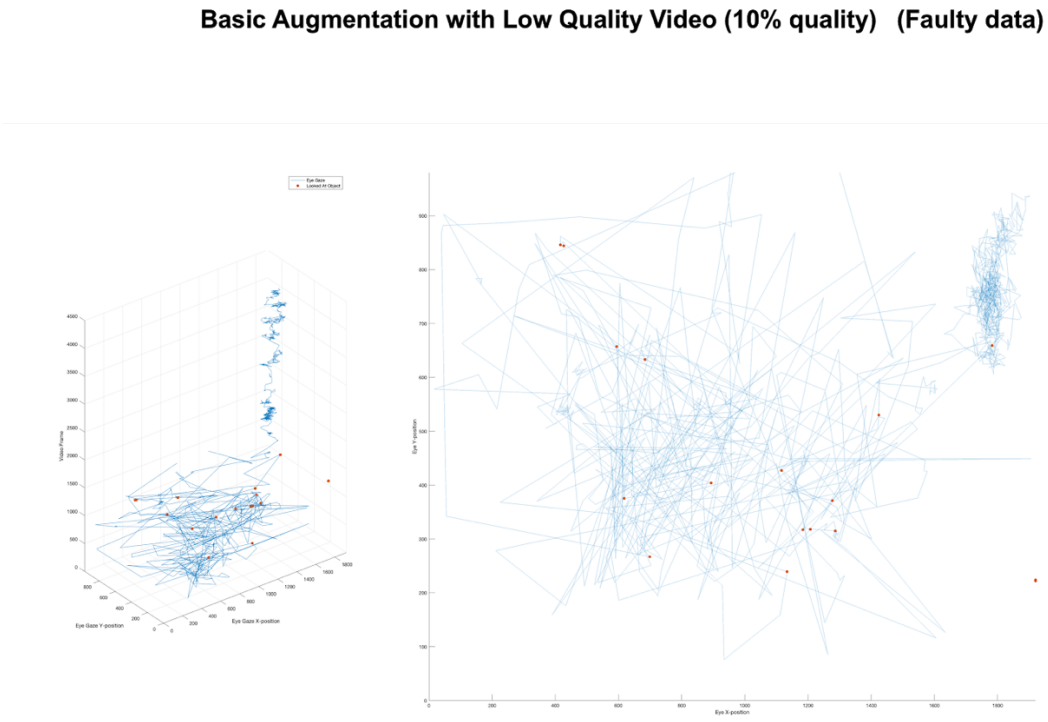



**Figure 25: Raw data obtained during the experiment that contains an error and was discarded from the set. The blue lines illustrate the eye-gaze path. The red dots indicate frames in which the participant gazed at an object.**

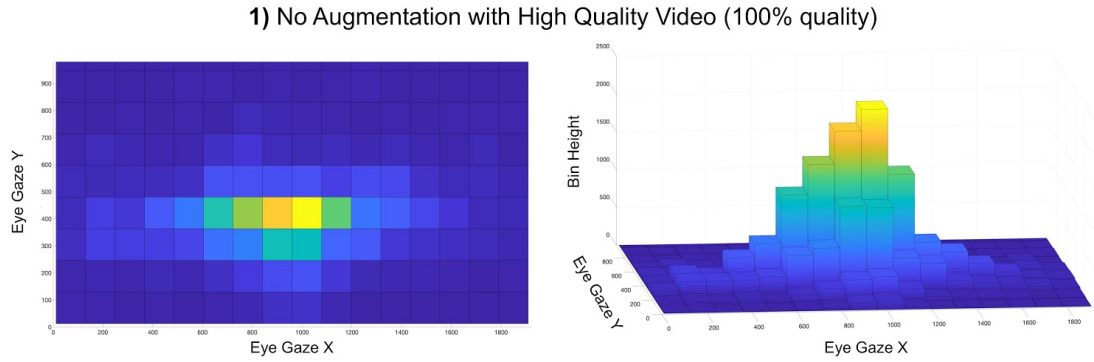



**Figure 26: Histogram displaying all gaze-readings obtained during the first section of the experiment.**

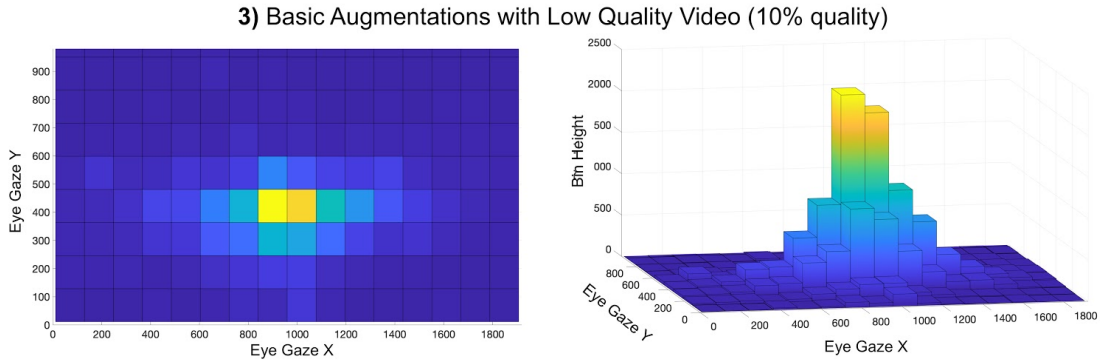



**Figure 27: Histogram displaying all gaze-readings obtained during the second section of the experiment.**

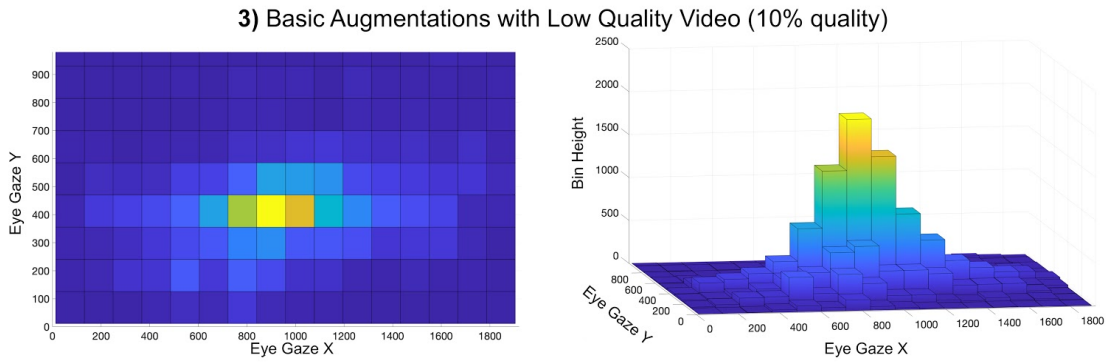



**Figure 28: Histogram displaying all gaze-readings obtained during the first third of the experiment.**

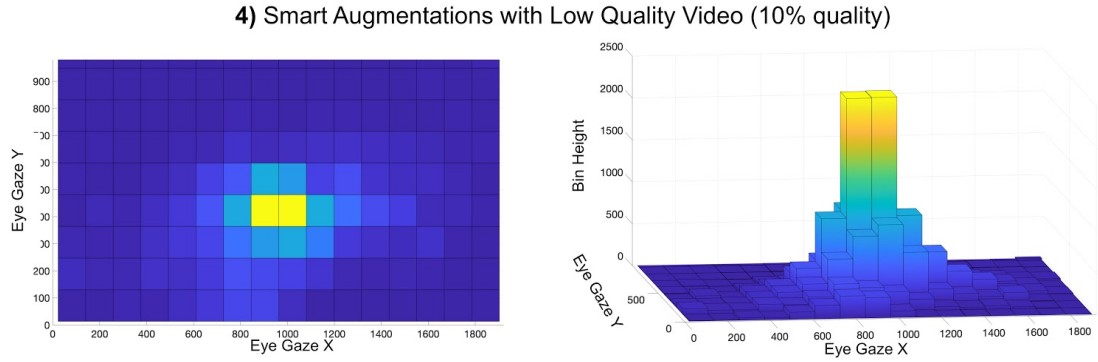



**Figure 29: Histogram displaying all gaze-readings obtained during the fourth section of the experiment.**

Although these histograms give some visual insight into how eye gaze behaviour varied across each of the augmentations, it does not yet show to what extent. To gain insight into this, it was decided to take the analysis of the data one step further.

The first step taken was to normalize the data across each of the augmentations. Now, based on the normalized data a 3D surf plot can be created which can be used to express the probability distribution that a gaze measurement fell in a certain probability region. Then, by taking a contour plot at various intervals, a map can be made of how much a certain probability maps out a region on the screen. The chosen probability intervals were selected based on experimentation. 10% was the highest probability that still generated a measurable area whereas 0.5% was the lowest probability that did not completely cover the screen, incremental steps of 0.5% were taken in between these bounds. Then, by taking a large sample of random points, and testing the coordinates falls within the bounds of a probability area, it can be calculated what fraction of the screen is covered by this specific probability area. The total number of random coordinates chosen was N=100000. Increasing this number any further did not improve the accuracy of the sampling method.

For each augmentation, the results can be seen in figure 29-34. The markings at which the smart augmentations are triggered have been overlaid for reference. Note, this area marks the driving lane from the perspective of the camera. during the video the driver's vehicle stood in front of a traffic light waiting for it to turn green. This might have caused the participants to focus on this more than during (1) and (4). Furthermore, it was assumed that for (4), the participants were more focussed with gazing around the centre of the screen, as this was the area from where the augmentations could also be observed. Unlike for (3), the augmentations were only visible during (4) once the object was near the centre of the screen as well.

When comparing the effects of the augmentations on the gazing behaviour of the participants, it becomes clear both have rather opposing effects. For the “basic augmentations”, the width of the gazing behaviours is the widest when comparing it to the others. As for the “smart augmentations”, this visualization illustrates how much focus there is on the centre of the screen, where the “smart augmentations” are being applied to it. The sides of the screen - where no augmentations are being applied - is barely looked at all. In figure 30-33, the surface areas of each probability surface can be compared.

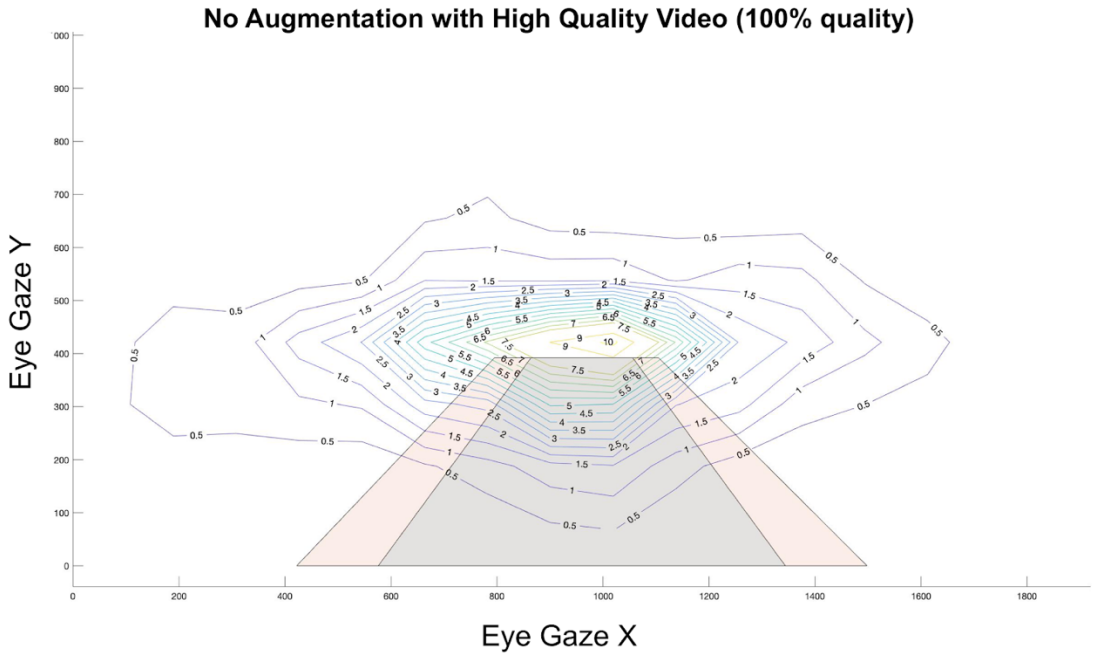



**Figure 30: Probability density areas displaying all gaze-readings obtained during the first section of the experiment.**

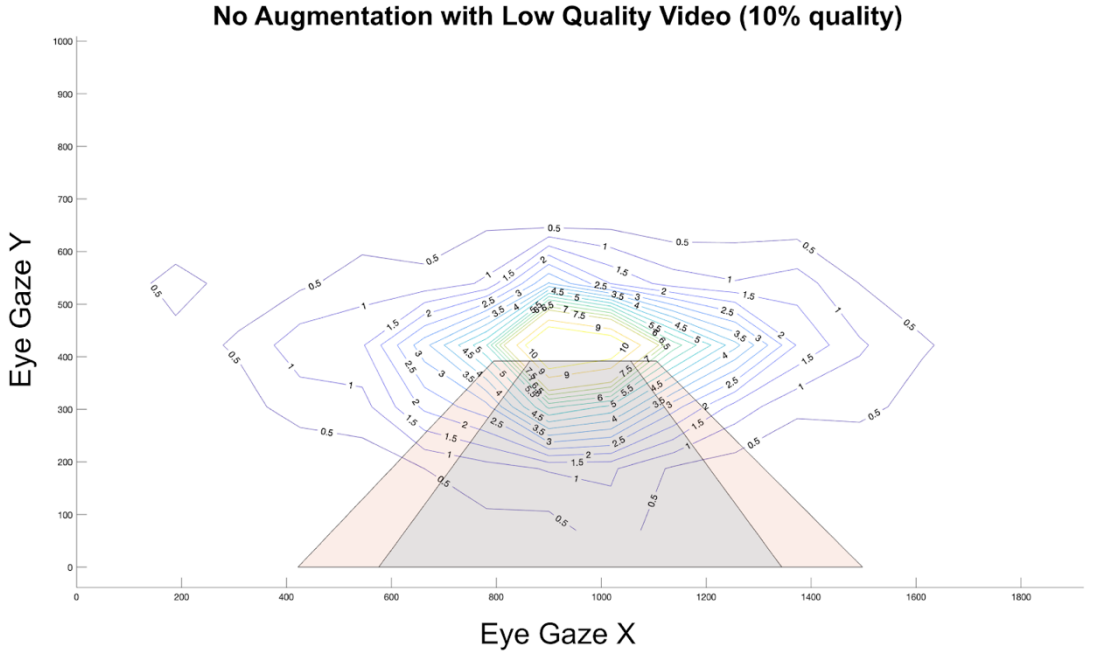



**Figure 31: Probability density areas displaying all gaze-readings obtained during the second section of the experiment**

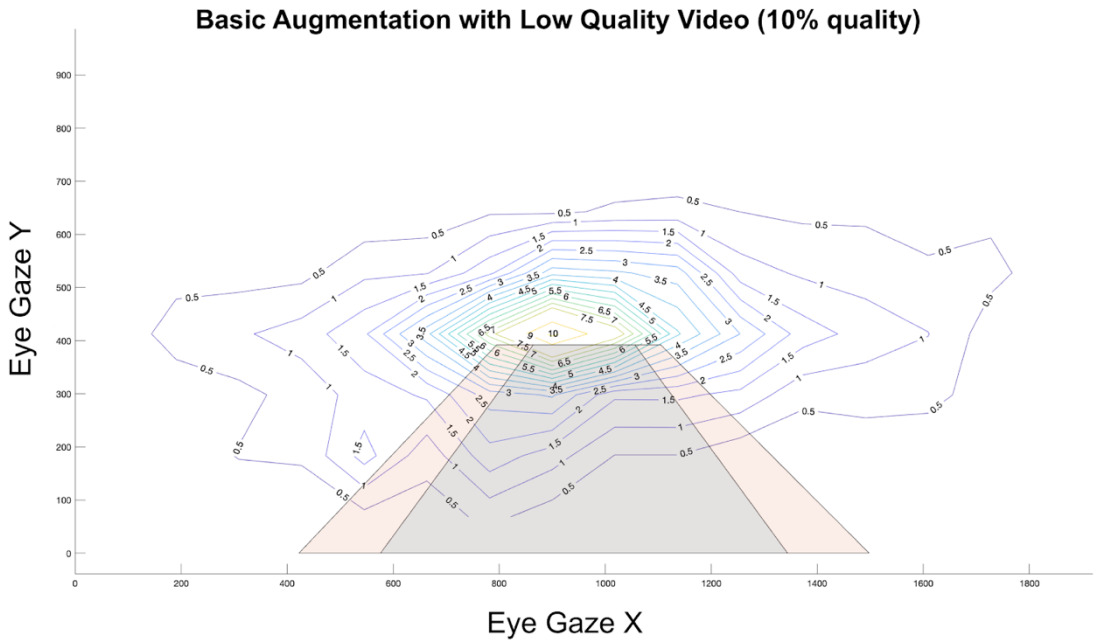



**Figure 33: Probability density areas displaying all gaze-readings obtained during the third section of the experiment.**

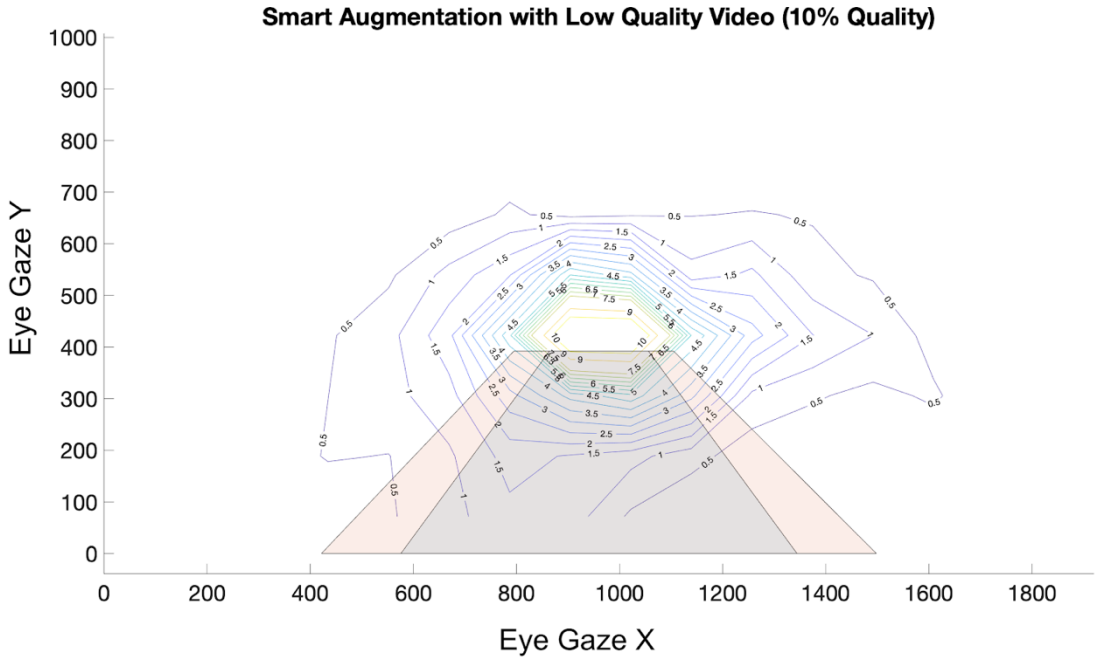



**Figure 32: Probability density areas displaying all gaze-readings obtained during the fourth section of the experiment.**

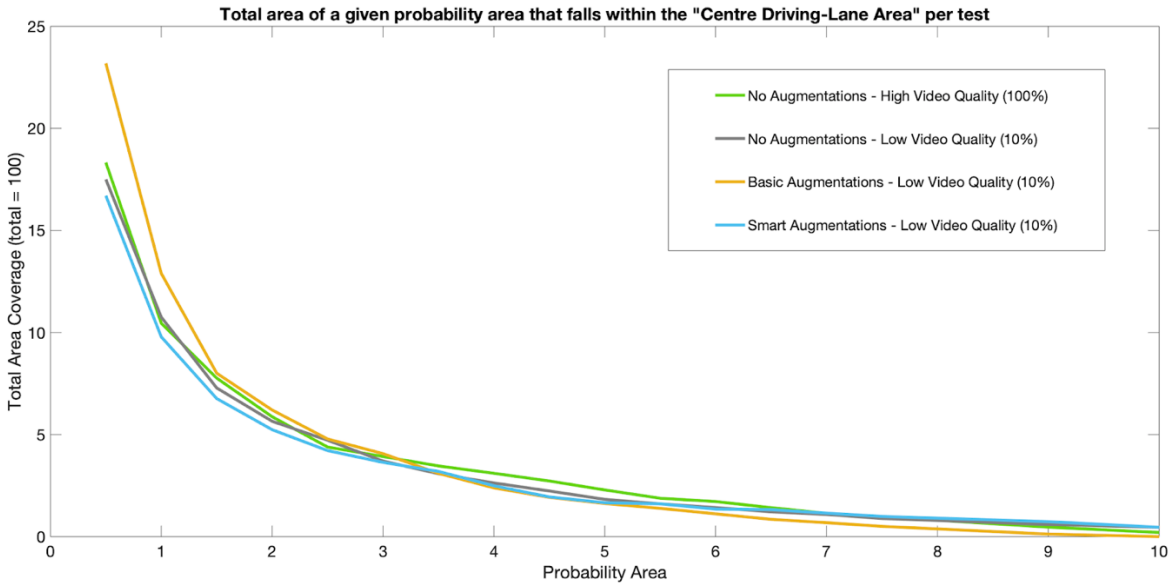



**Figure 34: Total area coverage per probability density area for a given eye-gaze percentage per section of the experiment.**

## 7.3 Recognizing Dangerous Events

Besides the eye gaze movement, the simulator also kept track in which frame participants spotted danger. Participants marked this moment, by pressing the space bar on the keyboard in front of them. Unlike the gaze behaviours, all datasets were deemed useful, as this part of the analysis did not depend on the eye gaze readings.

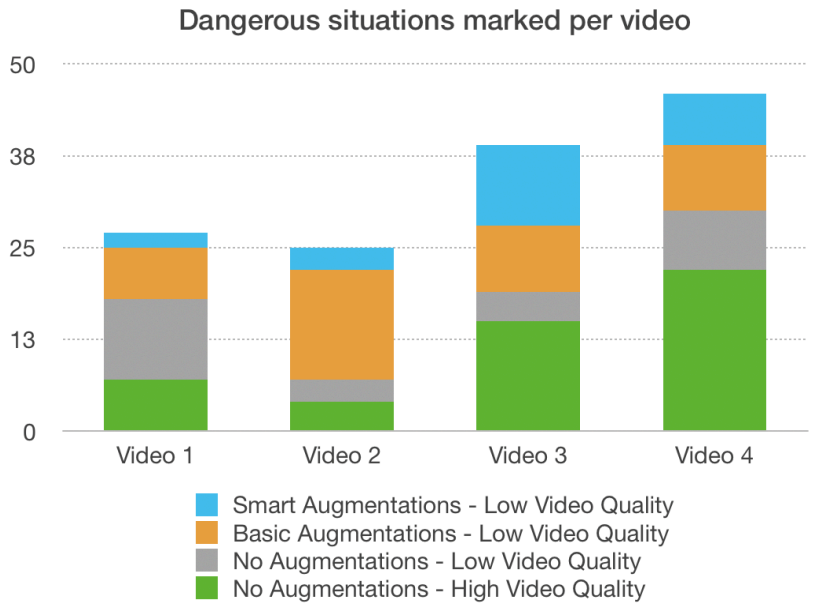



**Figure 35: Bar chart illustrating per video and section the amount of times a "dangerous event" was marked by a participant.**

In figure 35, an overview can be seen on the times per test in which the participants marked a dangerous event. What stands out, based on this data, is that the video and augmentation combination that has the most dangerous moments tends to be different depending on what type of augmentation was applied to the scene. In other words, video 2 seems to look a lot more dangerous with basic augmentations.

However, this is also where the subjectivity of the participants comes to light. Based on a closer inspection of the data, it was found that while some participants only marked just one or two events as dangerous per video, one participant marked up to eleven per video. Bearing in mind that the participants cycled through the videos in chronological order, it becomes apparent why this cycle of "most dangerous moments" changes through the data. Although all participants, that were recruited to participate in this study, were from the same country or origin (China), it did not equalize the danger perception as much as initially hoped. Therefore, to gain more insight into how the augmentation effects affected how well it increased (or worsened) performance of the participants in spotting dangerous situations, a different analysis approach was chosen.

By taking the participants' performance for each video, where participants watched it in a high video quality without any augmentations as a baseline, the effects of a lower video quality - with or without augmentations - can also be compared. Here, performance is defined as: 'how well were participants able to naturally spot danger in the scene where their performance is assumed to be the baseline' which can be compared with the other results. The comparison is done on an investigation into which participants saw the danger first (e.g., the baseline vs. low video quality and smart augmentations) and possible cases in which danger was not seen at all. First, it was manually investigated for which events in the video multiple participants marked it as a dangerous event. By doing so, various scenarios have been identified that were then further investigated. In Appendix B, some of these events are highlighted as examples. Through this approach, three general trends were observed, namely:

- First, participants that saw the video fragment in the low video quality - without augmentations - performed worse compared to all other groups of participants seeing the same scene in different conditions. In some cases, the danger wasn't even noticed at all. What also stood out, was that this group had various events marked that can be assumed to be false positives as well;
- When augmentations were applied to the video scene, participants tended to notice and mark the dangerous events sooner than participants who saw the same video fragment without any augmentations in a high video quality;
- When seeing the scene with the smart augmentations applied to it, the location at which the danger appeared on the screen started to matter a lot. Participants tended to notice dangerous events the fastest as long as the danger appeared near the centre of the screen. Contrary to this, for example when a pedestrian jaywalked onto the street appearing from the side of the screen, this group tended to perform generally worse. For the "basic augmentations" this effect was less noticeable, as the participants also tended to notice danger coming from the side of the screen, which is also in-line with the data previously obtained.

One possible explanation to why participants performed better at detecting danger near the centre of the screen, is that this is where their visual attention was focussed. Namely, participants tended to focus a lot more on the centre of the screen once the "smart augmentations" were applied to it. Therefore, it is likely that they were focussed on this area, at the cost of scanning the periphery of

the screen. In other situations, where the dangerous event starts away from the centre (e.g., a pedestrian jaywalking coming from the pedestrian lane), participants tended to be more delayed in noticing the danger. Although this could be caused by the participant's focus on the centre of the screen as well, there is also another possible explanation. Given that the borders that were being applied shifted to red, it might have caused the participants to wait for this border to become red before signalling the observed danger. When the danger appears at the centre of the screen, it likely already got a red border from the beginning, and therefore did not result in a delayed reaction.

## 7.4 Discussion

Based on the obtained results, there are a few noteworthy topics to expand upon. First, the data provides insight into how remote drivers or observers are impacted by a low-quality video stream. Second, there are findings on how augmented information can be served to them to improve their performance in assessing danger in remotely monitoring for danger. Third, there are also findings from these results that shed light on how the augmentations that are served to the user can be further improved.

As expected, participants performance was at its lowest the video scene was seen in a low video quality without any augmentations applied to it. As details disappear from the scene, it becomes harder to spot smaller objects such as pedestrians crossing the street. It should be noted that driving or monitoring in such conditions does not necessarily reflect a realistic scenario. In a more realistic remote driving/monitoring scenario, the video quality would only occasionally drop from time to time due to limited bandwidth availability or simply a bad connection. Therefore, the video would shift between various levels of quality depending on network conditions. Still, these results give interesting insights into how much performance can drop as a result: from slower reaction times in noticing pedestrians crossing the street to not realizing they are present in the first place. It also raises the question if a different type of augmentation can be applied to a video stream depending on the severity of the network disruption. For example, the video stream could show no augmentations as long as the video stream conditions are sufficient, but dynamically switch once the quality drops below a given standard. These augmentations could then be used to compensate for the loss in quality and highlight the positions of pedestrians and vehicles.

Based on the data that was obtained from playing the videos using two types of augmentations, a few conclusions can be drawn. One expected takeaway is that simply applying bounding boxes to all objects in the scene is not effective. It likely causes too much clutter all around the screen and participants do not perform better compared to when they see a scene in a low video quality without augmented borders. However, what is improved in participant performance, is spotting occasional smaller objects (such as pedestrians) which would otherwise go unnoticed. Therefore, a case can be made for even applying this type of not smart augmentations to a scene to increase the ability of the observer to disambiguate between objects in a scene.

One unforeseen finding was that the "smart augmentations" would result in a decrease in scanning behaviours. From a case-by-case analysis, it indicates that these augmentations are helpful for spotting danger around the centre of the screen, while worsening participants' performance at the periphery. Although further research would be required, it raises the question whether the overall performance can increase if augmentations would only be applied based on where the participant would be gazing, rather than at a fixed location on the screen. By doing so, the participants view would not be guided as much and still allow them to freely scan the scene while having access to the same information-layer augmented to it.

# 8 Future Work and Conclusion

In this section, suggestions for future work and the overall conclusion for the first part of the thesis are presented.

## 8.1 Future Work

There are multiple possible expansions that would improve this body of work; this section aims to discuss some of the possible angles that could be taken into consideration when expanding upon this work. Two directions are suggested in this section; future work for improving the simulator and expanding upon the AR capabilities.

### 8.1.1 The Simulator Environment

So far, most research efforts have been put into creating the simulator environment alongside testing the effects of a few basic augmentations. Due to time constraints, minimal effort was put into involving user feedback into how the augmentations should be displayed. Future research work could investigate in what manner this information could be served through AR most effectively. Doing so would likely significantly improve the user experience of the augmentations. For example, what sort of design suits best for warning: should the notification be displayed on the object itself, or would the driver have more benefit with a warning message in a fixed location. Other modalities could also be explored, sound or embedded light interactions are also possible future directions to explore.

Besides expanding the research that has been done thus far, the Test Bed itself could also be further improved for research purposes. Another consideration for further improvements would be to integrate more types of measurements w.r.t user perception of the augmentations. One example could be to add physiological sensors such as galvanic skin response (GSR) so that factors such as stress can be measured as well. This could give more in-depth insights into how the AR messages are being displayed.

### 8.1.2 The Augmentations

One comment that was frequently by some of the participants, was that it would be very helpful to have an augmentation that indicates lane lines for when the video quality is in a bad state. Although the current augmentations give an overview of some entities in traffic, it is not enough to accurately judge if the vehicle is driving appropriately.

Other types of augmentations could also be explored using the testbed. For example, displaying the intention of the vehicle by augmentation the lane in which the vehicle is going to be driving in would likely add great value to the experience. Other possible augmentations would be to display information with regards to both the environment and the vehicle itself, such as the speed limit and what speed the vehicle is currently driving at. Figure 36 showcases what these augmentations could look like.

Additionally, during this research, focus was put on testing interactions where AR provides information without the user specifically requesting it. Another interesting angle would be to create the interactions as such so that they appear once the user gazes at an object (e.g., showcase time to

green at an intersection) or warn of danger that the user hasn't observed yet and disappears once noticed.

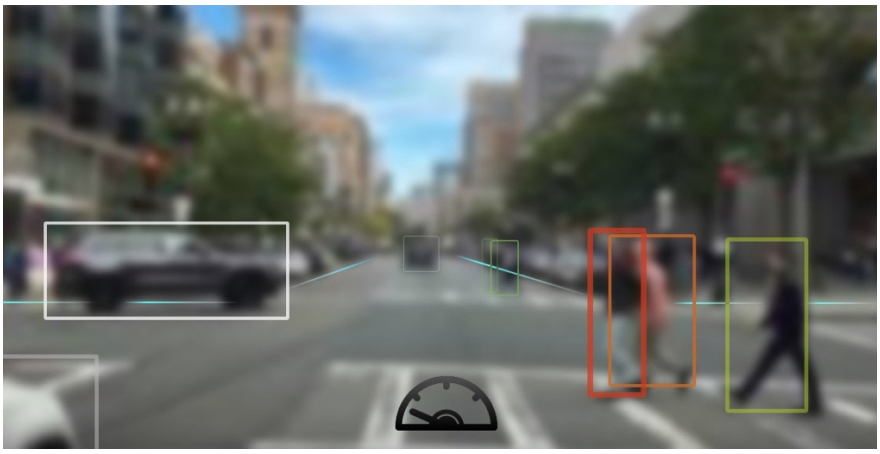


**Figure 36: Envisioning of a future work direction in which the sides of the road are also augmented, as well as a speed-meter in the bottom of the screen.**

### 8.1.3 Human-trained danger detection

Another direction for future work that was discussed -but not explored in this work - is that the simulator environment could potentially be used to obtain training data for a machine-learning model for "danger-classification". By collecting a large dataset of research-participants observing the vehicle driving through the city (while scanning for dangerous scenario's), a dataset could be obtained to train such a classifier based on their eye-gaze readings. In this model, it would be assumed that the areas where most gaze-points are, are the most critical areas and objects that should be payed most attention to by the remote observer. Then, this classifier could be used to augment those areas and objects to guide the gaze of the remote observer.

To validate if such a classifier would be useful in the context of remote driving, a Wizard of Oz demo could be created first. To create such a demo, a much smaller dataset is needed, which is a huge benefit (possibly, it can even be curated by the researcher(s)). For this demo, no ML-model would be necessary: objects that were frequently gazed during the "training-stage" could be augmented by a rule-based system without making use of ML.

## 8.2 Conclusions

In the first part of this thesis, it was investigated if AR could be used to compensate for a reduced video quality during the remote observation of a remotely driven vehicle. A simulator environment was created that made use of eye-tracking technology to measure the effects of the augmentations on the video stream presented to the observer. This simulator was capable of playing videos of a vehicle through a city, as well as applying augmentations in real-time to these videos. These augmentations made pedestrians and vehicles stand out more from their environment. The simulator was used in a user test to evaluate two different applications of Augmented Reality.

As part of this research experiment, participants were exposed to four different scenarios in which their task was to monitor a remotely driven vehicle. Based on collected data, eye gaze behaviors

across participants for every scenario were compared, as well as their ability to spot dangerous events across these scenarios.

The first case served as a “control case” where no augmentations are applied to the video, and the video is shown in high video quality. In the second scenario, the video quality was severely reduced. From the findings, it appeared, as expected, that participants had a harder time recognizing the danger, and they needed to scan the video more intensively. In the third scenario, a “basic” augmentation was applied to the video in real-time through object detection techniques. As part of this basic augmentation, pedestrians and vehicles were given a bright colored border to make them stand out more from their environment. The video was shown in a reduced quality, just like the second scenario. What the data showed was that even this early-stage implementation improved the participants' ability to recognize dangerous scenarios. However, participants still had to look around more extensively compared to the base-case. In the final scenario, a “smarter” augmentation was applied to the video. With this type of augmentation, only objects near the center of the vehicle where being augmented. This significantly improved the ability of the participants to recognize danger.

What stands out from these findings is that augmented reality can be used to compensate for a drop in video quality when it comes to recognizing dangerous events to some extent. However, it should be noted that it can also possibly lead to non-beneficial effects as well, such as that observers no longer check-in places where augmentations are not being shown. For future work, this could be a potential area to explore in further detail. The simulator that was developed as part of this research can serve as a platform for future work. Due to the way it was built, the augmentations could also be tested in an actual remote driving scenario.

# 9 The First 5G Enhanced ADAS In A Real Vehicle

This section of the thesis covers: the work done to build a demonstration of various ADAS use cases enhanced through 5G. The first part covers the ideation and overview of these use cases. The second part gives an overview of the teams and companies that collaborated on this project, as well as the planning and logistics for how the demo was showcased at the event. The third part discusses the implementation of all the components developed as part of this thesis. Finally, the outcomes of the demo, the media in which it was discussed, and possible future works are covered as well.

## 9.1 ADAS use cases

Four different ADAS systems were developed. Amongst these systems, four demonstrations of V2V (vehicle-to-vehicle) and three V2X (vehicle-to-everything) cases. The concepts behind these ADAS implementations were based on a brainstorm held by KTH and Ericsson. The topic in focus for the brainstorm was the available technology for the demo, and how these could be applied to create compelling use cases (figure 37). An overview of the data-flows for each use case can be seen in figures 37-40.

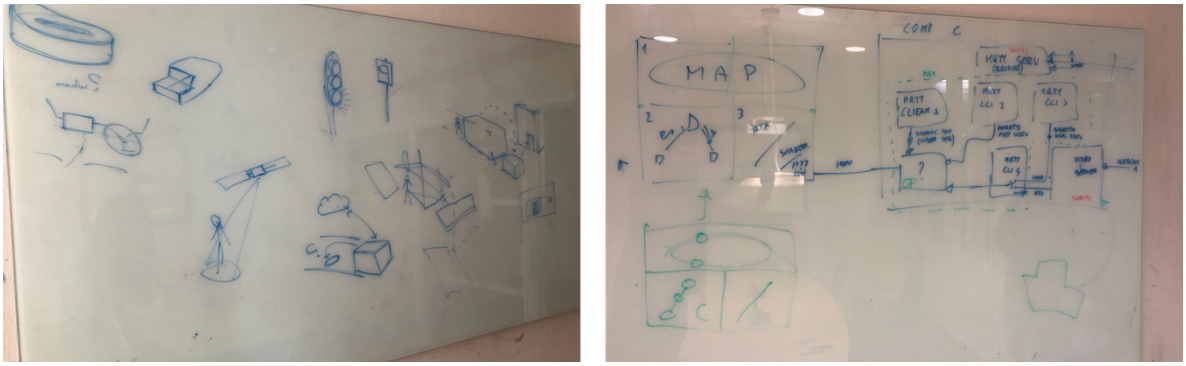



**Figure 37: Pictures taken from the whiteboard during the various brainstorm sessions that were held during this part of the project.**

### V2V – Aquaplaning (figure 38)

This use case demonstrates a sensor sharing scenario between two vehicles. The first vehicle detects the beginning of aquaplaning (1), and sends out a warning (2). The vehicles approaching this location then will get a warning message (3). On the HMI of this vehicle, the lane where the danger has been detected will be augmented. Based on this information, the driver can either slow down or, if possible, change to a different lane.

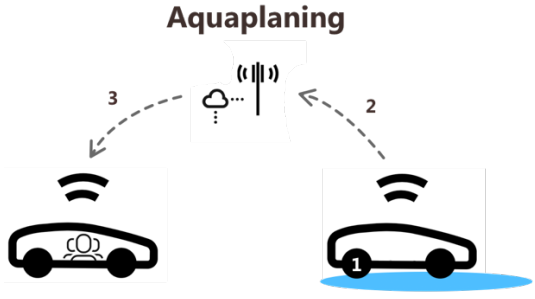



**Figure 38: Illustration of the data-flow in the aquaplaning use case.**

### V2V and V2X - Road Incident (figure 39)

For this use case, the first vehicle is involved in a road incident and needs to break urgently. Upon the emergency brake, a warning broadcast will be sent out (1). The vehicles approaching this location then get a warning message (2). On the HMI of the upcoming vehicle shows a warning message related to the just happened accident in front of them and in how many meters it can be expected to encounter it. The second part of this use case entails the beginning of a video stream to the authorities from the second vehicle (3). Based on this information, it can be investigated if emergency services, such as an ambulance, needs to be sent to the scene (4). Therefore, this part of the use case could also be considered a V2X use case.

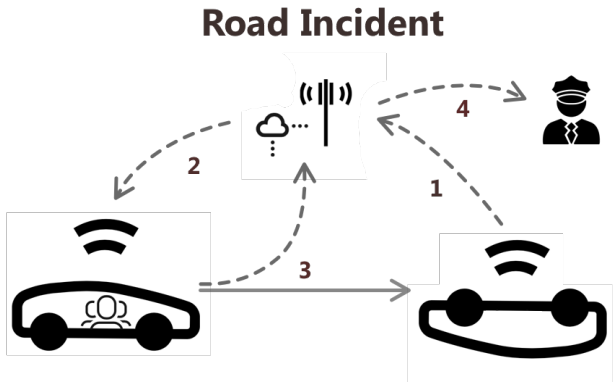



**Figure 39: Illustration of the data-flow in the road incident use case.**

### V2X - Sign Translation (figure 40)

This use case involves a situation where a driver is confused by a traffic sign and aids them in understanding it. For example, when driving in a foreign country that has a different standard for traffic signs. The system inside of the vehicle detects that the driver is trying to understand the sign by using eye-tracking (1) (e.g., by staring for too long at the sign). Upon detection, the sign is translated through a cloud-service (2) and its translation is shown on the HMI of the vehicle (3).

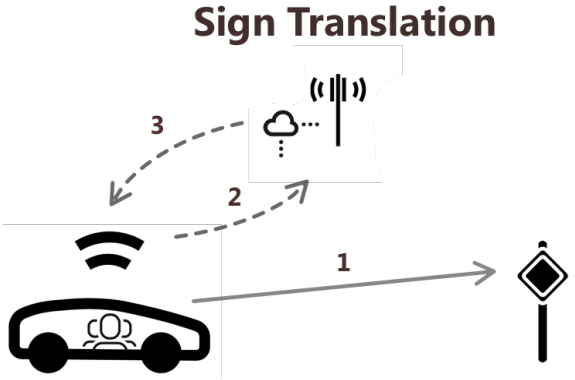



**Figure 40: Illustration of the data-flow in the sign translation use case.**

**V2X - VRU Detection (figure 41)**

Consisting of two separate components, this use case demonstrates the detection of out of sight vulnerable road users (VRU), who are being distracted by their phone. Both parties have been sending their GPS locations to the cloud (1). The first component focuses on alerting the driver that a pedestrian, who is currently out of the line of sight, will soon cross the vehicle's path (2). For the pedestrian, a warning will appear on their phone that a vehicle is approaching as well as from which direction it will approach (2). Then, once the pedestrian has been detected by the driver - confirmed through eye-gaze (3) - the pedestrian receives notifications that they've been seen by the driver (4).

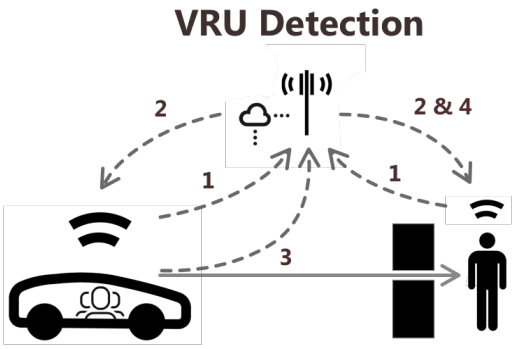



**Figure 41: Illustration of the data-flow in the VRU detection use case.**

## 9.2 Planning and Logistics

### 9.2.1 Collaborative Partners

In order to make the demo feasible, responsibilities were divided amongst the participating companies and their respective teams. This section aims to give an overview of the responsibilities each partner took upon them.

Audi provided the vehicles to be used for the demo: an A-8 and Q-8. The Q-8 was selected as Vehicle-A, while the A-8 was used for Vehicle-B.

Ericsson provided the interface for Vehicle-B by sharing the data through MQTT. Over this interface, position data and aquaplaning measurements (from the Pirelli-tires) were sent. Additionally, Ericsson took on responsibility for overall project management. Besides coordination between partners, this also included responsibilities such as ensuring that the Lingotto rooftop track was available.

Italdesign Giugiaro took on the responsibility to ensure that the hardware integration within both vehicles was taken care of. For Vehicle-A, this included the power supply and fixture for the laptop, the camera near the rear mirror, and a connection to the HMI so that the laptop could take over display-functionalities on the onboard HMI. Additionally, they provided the visual-designs for the interface elements to be displayed on the HMI in Vehicle-A. This was done to ensure a seamless integration between the HMI design and Audi's brand design requirements. These designs were based on low-fidelity mock-ups provided by KTH. During the demo, they also provided the crew of Vehicle-A (both driver and tour-guide), as well as the actor for the VRU-role.

Pirelli provided their cyber-tires, which are capable of detecting various levels of aquaplaning. These tires were installed on Vehicle-B (the Audi A-8). Furthermore, to ensure that aquaplaning would be possible on the roof, they engineered solution to ensure that water would stay on the roof (as much as possible). To broadcast the detection of the aquaplaning, they collaborated with Ericsson. They also provided a certified driver during the event; given that the driver needed to drive relatively fast while aquaplaning on a rooftop, this came in handy.

TIM (Telecom Italia), provided the 5G-coverage for the Lingotto track. In collaboration with Qualcomm, they ensured that the 5G-coverage was sufficient for the demo. Qualcomm also provided 5G-routers that were installed in both vehicles.

Tobii Pro, provided three sets of their Tobii-Pro glasses to be used for the demo. They also provided hardware such as power banks.

The primary responsibility of the author (KTH), in regard to technically implementing the planned use cases, was to provide an integrated HMI-experience for Vehicle-A. What this included was the following:

- Ensure that the fidelity-level of the HMI was on “production-level,” based on the designs provided by the designers from Italdesign Giugiaro;
- Develop the sign-detection and classification pipeline;
- Develop the VRU-detection and augmentation pipeline;
- Develop AR-augmentation effects for the aquaplaning notification;
- Communication of the status of Vehicle-A with Laptop-C, so that the correct information could be displayed on the screen.

In collaboration with Pietro Lungaro (KTH), other components were taken care of, such as setting up a geo-fenced area, used for triggering certain events during the demo, as well as planning out the overall architecture of the demo set-up. Furthermore, Pietro Lungaro took the responsibility of creating a dashboard displaying all events on a map and providing additional information in the visitor area of the demonstration.

### 9.2.2 (Remote) Collaboration

Given that the teams were situated in different countries (e.g., Audi - Germany, KTH - Sweden, IDG - Italy), weekly meetings were arranged over Skype to keep the teams synchronized on the processes. Teams would follow-up with each other outside of these meetings, in case collaboration between project members was necessary. An example of such collaboration happened during the design of the HMI, where KTH provided IDG with a set of mock-ups that were then branded and styled by IDG, to be technically implemented by KTH.

Before the demo, there was one opportunity to meet-up in Turin and attempt to integrate some of the systems already. Various challenges appeared during this meeting. First of all, time was minimal, with only two days available for testing. Only one of the two vehicles were available (Vehicle-B), and the Pirelli-tires were not installed yet. As we were unable to test on the rooftop of the Lingotto building on the first day, most tests were performed on a nearby parking lot. Most of these tests were to confirm various components, such as the power supply installed for the high-performance laptop and internet-access through the Qualcomm modem.

One week before the demo, all team members and material were present to test and refine the demo set-up over the course of the week. Most of the critical components were verified prior to the demo, but full integration had not been tested before. For the first three days, the most focus was put on implementing, testing, and making each use case more robust. In the remaining days, most time

was spent refining the demo with regards to the timing of certain events, adding geo-location triggers, and improving the overall story of the demo. The main challenge faced during these final days was balancing the engineering-efforts versus the pr-efforts as multiple filming sessions had been planned. These sessions collided with the much-needed time for ensuring the demo would run smoothly. In the end with a combined effort from all involved parties, all components were implemented in time.

### 9.2.3 Spatial Arrangement of the Demo

This section aims to provide an overview of how the different use cases were arranged along the Lingotto rooftop.

When it came to arranging the different use cases spatially, a view key considerations had to be taken into account. First, the aquaplaning use case could only be performed on specific parts of the rooftop due to the challenges in controlling the water-level, and due to the water splashing off the rooftop. Given that the lower floors of the Lingotto building are occupied by a mall, a location was picked so that the water would not splash down onto innocent passersby visiting the mall. Additionally, Vehicle-B needed to achieve a certain speed unless no aquaplaning would occur and therefore no detection either. Therefore, the vehicle needed a certain runway to get up to speed as well.

For the sign-detection use case, there was a soft-requirement regarding the position of the sun. If the sun were to shine directly onto the sign, there was a risk of the classifier being unable to detect the sign. Prior testing in Stockholm, proved to be no issue, but during the testing in Italy, the brightness proved to pose a challenge. Therefore, it had to be placed so that it got minimal exposure to direct sunlight. This was done to increase the robustness of the classifier.

Finally, a requirement was set that the demo should not last for more than 5 minutes, including on- and off-boarding of passengers. It was assumed that this procedure would take about one minute, given that the passengers were given a short introductory talk by the guide on board of the vehicle prior to the start of the demo. As for on- and off-boarding the passengers, this preferably had to happen close to the tent where the demo was being presented.

All of these factors resulted in various trade-offs that had to be made. In the end, an arrangement was proposed that satisfied all the hard-requirements and took most wishes into account. Figure 42 provides an overview of the location of where each use case happened.

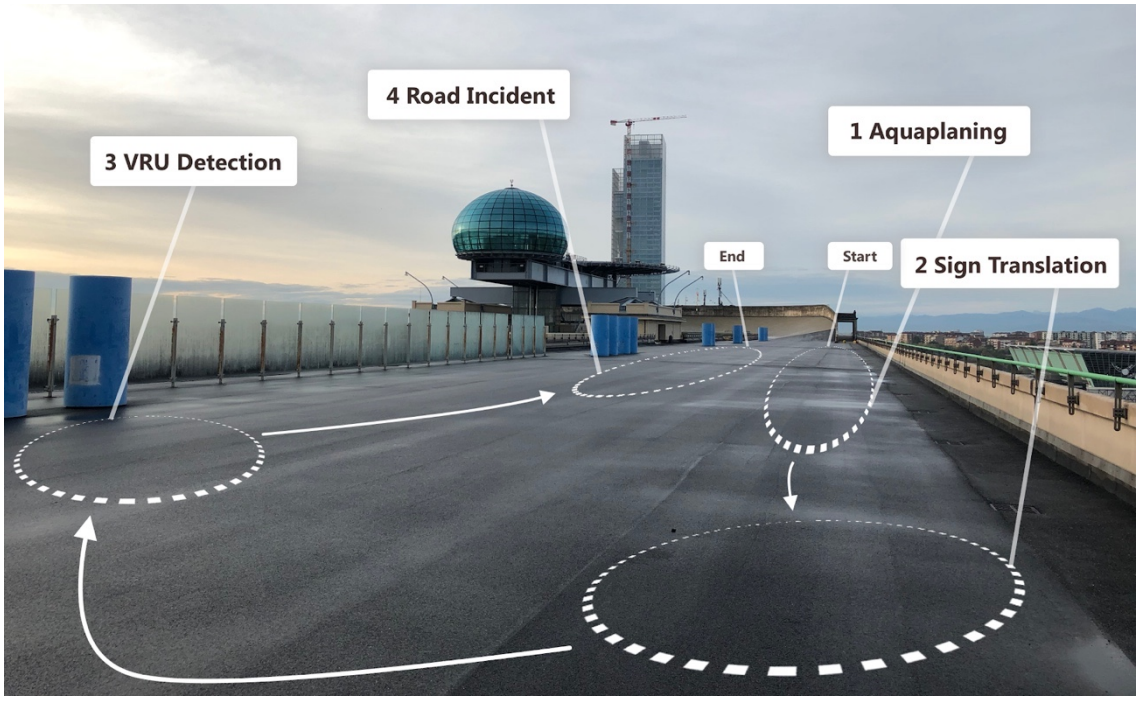



**Figure 42: Illustration of how each use case was spatially arranged on the rooftop of the Lingotto building in Turin, Italy.**

## 9.3 System Implementation

The following section will discuss various parts of the technical implementation done by the author for the 5GAA demo. This section mainly aims to cover some of the strategies and techniques used to develop the demo.

### 9.3.1 Design and implementation of the user interface

Based on the use cases, various concepts for the user interface of the onboard HMI were developed. These were discussed with Italdesign, who refined the designs to be in line with Audi's brand standards. Figure 43 presents the overview of the user interface as these were implemented in the demo.

### 9.3.2 Total System Overview

In figure 44, a total overview of the system, and all its technical components, are shown. In total, the Torino Demo required three laptops. One laptop was installed in each of the vehicles, and one laptop was used for displaying a real-time map of the rooftop displaying where each vehicle was at a given time. In both cars, 5G antennas were installed so that internet connectivity could be provided. The laptop displaying the map received internet by tethering with a 5G smartphone.

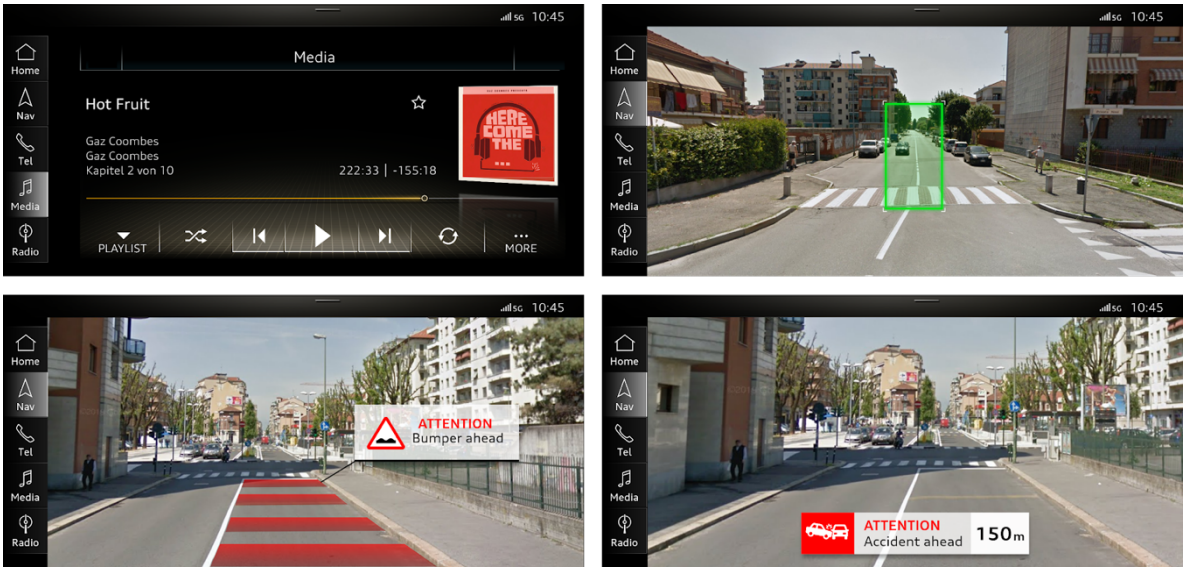



**Figure 43: Designs for the User Interface of the HMI provided by Italdesign.**

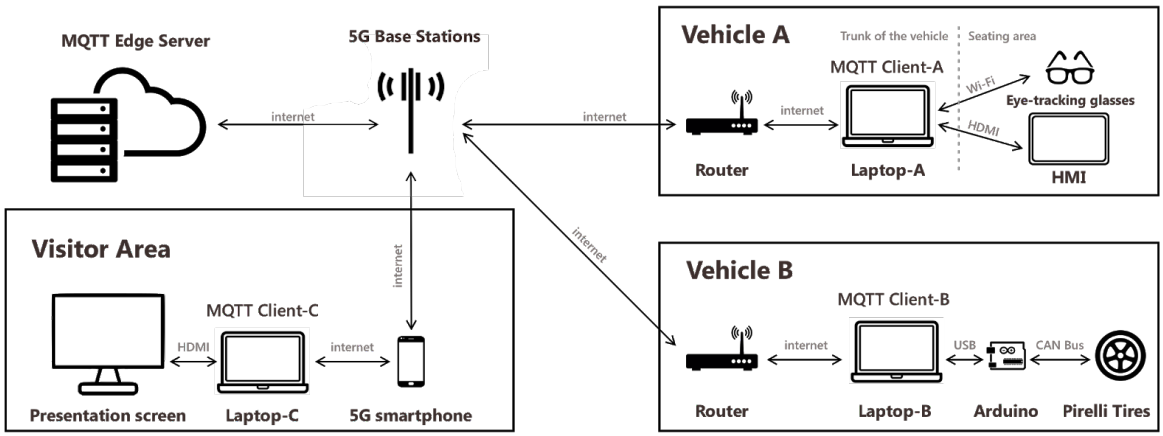



**Figure 44: An overview of all the technical components used within the Torino Demo.**

### 9.3.3 Hardware

Compared to the simulator, discussed previously in the thesis, this application ran on slightly more powerful hardware. The developed application was running on an Alienware 17 R5 laptop, running the Windows 10 operating system and equipped with 16GB of installed RAM, an Intel Core i9 CPU, and an NVIDIA GTX 1080 GPU.

The application was displayed on the onboard HMI of the vehicle. Italdesign modified the vehicle so that the onboard system could be bypassed and so that the Alienware computer could directly connect - through HDMI - with the display. The computer itself was securely mounted in the trunk of the vehicle (figure 44). A clipping mechanism, including cushioning, was installed so that the computer could be safely secured. To get access to the 5G network, the computer was connected to a prototype 5G-router from Qualcomm. The router and its shark fin antenna can also be seen in the figure. Powering the computer proved to be a big challenge: the vehicle itself was not able to provide

the power necessary due to the power-consumption levels of the high-end computer. Therefore, a power supply and transformer were installed in the trunk so that adequate power could be provided throughout the day. Finally, To measure the eye-gaze of the driver, a Tobii Pro Glasses 2 was used [37]. This is the same model of Tobii glasses as was used during earlier user research.

**Figure 45: The laptop that was running the application displayed in Vehicle-A was hidden in the trunk of the vehicle, alongside with the other hardware such as the router and power supply.**

### 9.3.4 Software

The interface for the HMI was developed in C++ using openFrameworks (OF) version 0.10.1 as its main framework. A few modifications have been made to OF specifically (for further information on these, please refer to section 4.1.2 of this thesis document).

## 9.4 Implementation of the use cases

### 9.4.1 Aquaplaning

The cyber-tires from Pirelli handled the detection of the aqua-planing. These tires would continuously publish a warning-level through the CAN bus of the vehicle. A laptop present within this vehicle would broadcast once the early stages of aquaplaning were being detected. This broadcast also included the GPS-location of the detection.

Upon receiving the message from Vehicle-B, Vehicle-A would trigger the Aqua-Planing use case. On the dashboard of the vehicle, a video-stream from the dashcam would be displayed, with a lane augmentation on top of the video. The lane-augmentation indicated in which lane the aqua-planing was being detected. The location of the lane could be calibrated apriori to the demo. Given that the pavement on the roof of Lingotto, best described as a single broad lane without markings, it did not allow to develop a lane-detector within the time-scope of the project, this approach was chosen.

The lane-augmentation consisted of a set of four vectors, one for each “corner” of the lane. Based on these vector-coordinates, a polygon was defined in which translucent red-rectangles would flow towards Vehicle-A. This flow of rectangles was created through a time-based animation. The height at which each rectangle was displayed was time-based; meaning that over time, the rectangles would descend towards the bottom of the screen. This created the illusion of a flowing stream of red-rectangles towards the vehicle on the HMI.

For creating each of the rectangles, the top-left corner was used as a reference point. The position of this rectangle was based on a mapping-function between the location of the top and bottom corners of the lane-polygon. In turn, the position of the top left corner of each rectangle determined the vertical-position of the other rectangle-corners. Their horizontal-position was determined by the line-vectors obtained by subtraction of the left and right coordinate-vectors.

Once vehicle-A had passed the location in which Vehicle-B had initially detected the aquaplaning danger, the animation on the dashboard stopped. Furthermore, an MQTT-message was broadcasted to notify laptop-C that this use case had ended.

### 9.4.2 Sign Translation

For detecting the traffic sign, it was initially assumed that the same neural network-based object detector, previously applied in earlier research during the thesis, could be reused. However, upon testing with the actual traffic sign that was designated to be used within the demo, it became apparent that the network was incapable of detecting the sign reliably. A choice had to be made; either retrain the neural network through transfer-learning with added training data on the specific sign or develop a classifier by hand.

Due to the approaching deadline, the decision was made to use a heuristics-based solution, rather than retraining the CNN through transfer-learning. The remainder of this section will discuss the implementation of this system.

By making use of the Tobii-glasses in a novel way, various shortcuts could be taken in the development of the classifier. First of all, the gaze-position of the user was already a given, as these were being provided by the glasses, as well as a current 'point of view' perspective from the driver of the vehicle. Therefore, this allowed for a crucial 'region of interest' filter. The classifier only needed to be capable of identifying the traffic scene in an area around the current gaze-position; rather than within the original image.

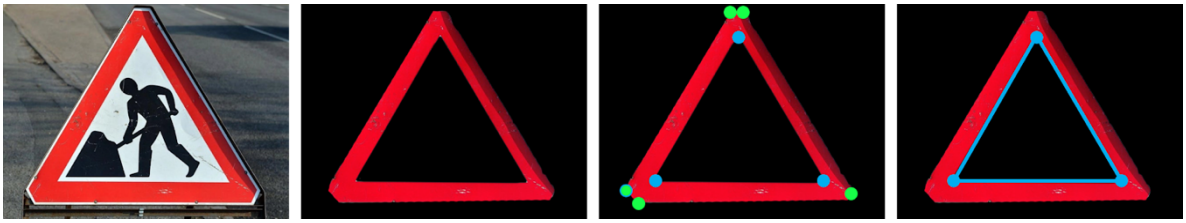


**Figure 46: Illustration of the steps in the sign-recognition pipeline: first the image is thresholded based on color values; then, the corners in the sets of contours are identified; the system then looks for a contour with three sides.**

To start this process, the current image-frame from the Tobii-glasses is obtained first. Then, based on the position of the eye-gaze, a subsection around the gaze-location is cropped from the image frame. This patch is the area of interest for the rest of the steps in the classification process:

- First, the color space was converted from an RGB (Red, Green, Blue) to HSV (Hue, Saturation, Value) color space. This allows for more robust detection of specific colors, such as red stroke around a traffic sign. This is followed up by two thresholding filters that only let red-pixels through. The output of this step is a binary Matrix where each non-red pixel is indicated with a 0, and every red-pixel is indicated with a value of 1.
- The second step is an eroding operation of achieving noise-reduction. The goal of this step is to remove individual small patches of red pixels from the Matrix. Each non-red pixel, as

obtained from the previous step, is increased in relative-size, by setting neighboring indices to 0. The output of this step is a binary Matrix where noisy patches of red-pixels have been set to 0.

- The third step uses a built-in OpenCV-algorithm to find contours in the Matrix. Each contour is a set of fitted-lines that in unison produce a polygonal shape. The output of this step is a set of contours of which each contour outlines a shape that has been identified within the Matrix.
- The fourth and final step is the classification step. In this step, for each contour-based shape, it is checked if the shape fits the description of a triangle, a shape that consists of three contours. For each shape that consists of three contours, it is assumed to be a triangle. However, there might still be some noise left in the image; and false-positives might still throw off the classifier. Therefore, a final check is done within this step, we only assume the triangle to be belonging to a traffic sign if it has a significant perimeter. Small triangles (assumed to be the results of noise) are therefore not considered. In case a triangle with a significant perimeter is identified, the confidence score is incremented.

This process is also illustrated in figure 46. Once this score reaches its ceiling (obtained by identifying a red-triangle in 20 subsequent frames), it is assumed that the wearer of the glasses is looking at the traffic sign. At that point, the user interface shows the translated sign on the HMI.

### 9.4.3 VRU Detection

For detecting the “vulnerable road user” (VRU), both GPS and object detection were applied. Given that within the demo, the location of the VRU was static (and wouldn’t change over the day), a geo-location was selected upon which the VRU-scenario would be triggered. Additionally, the VRU-scenario would only be triggered in case the car was heading in the right direction. Therefore, in the first half of the lap, the scenario couldn’t be triggered as the vehicle was approaching the sign-detection scenario.

Once the scenario started, an MQTT message was sent with the purpose of sending a warning to the VRU that a car was approaching. Within the car, the object-detecting was used to detect the VRU within the video provided through the dash-cam of the vehicle. Once they became visible, the VRU was then augmented on the dashboard of the vehicle by adding a translucent orange rectangle on top of them. Simultaneously the video, from the Tobii glasses, was also analyzed by the object detector. Once it was identified that the driver had seen the VRU, the translucent rectangle would shift to a green color. Once this happened, another MQTT-message would be fired, which in turn would show the VRU (on their phone application) that the driver of the approaching vehicle had seen them. After the driver had spotted the VRU, the scenario would end itself after a few seconds, giving the guide some time to explain what just happened.

### 9.4.4 Road Incident

Vehicle-B initiated the beginning of this use case. Given that an incident was being simulated, a button was pressed from within the car after the vehicle performed a hard-brake. This button-press activated the sending of an MQTT-message, which would then be interpreted as the detection of an accident.

For Vehicle-A, this implied that the Road-Authorities use case was ready to begin. Given that the vehicle was already streaming the video’s (dashcam and Tobii Glasses) to Computer-C, nothing had to change in this regard; Computer-C handled this by displaying the stream of both cameras on its monitor. As for Vehicle-A, on the dashboard, a video-stream from the dashcam was displayed, with

a warning message augmentation indication that an accident had happened a few meters in front of them. The MQTT-message provided by Vehicle-B also included the GPS location at which the button had been pressed. Once Vehicle-A passed this location, the scenario would end, and the dashboard would return to normal. At the same time, Vehicle-A would send an MQTT-message indicating that the use case had ended and that the stream no longer had to be displayed on the monitor of Computer-C.

# 10 Outcomes, Future Work and Conclusion

## 10.1 Outcomes

Based on our estimations, we assume over 200 participants were able to experience the demo during the day. The whole system was able to run nearly non-stop for the entire event without any unforeseen events. Only one crash occurred, but the issue was identified and solved within ten minutes. In the days that followed, the work was featured in various news articles that were released, including publishers such as Wired, la Repubblica, Automoto, and others [56, 57, 58, 59, 60, 61]. Figure 47 provides a visual overview of the four use cases that were demonstrated during the event.

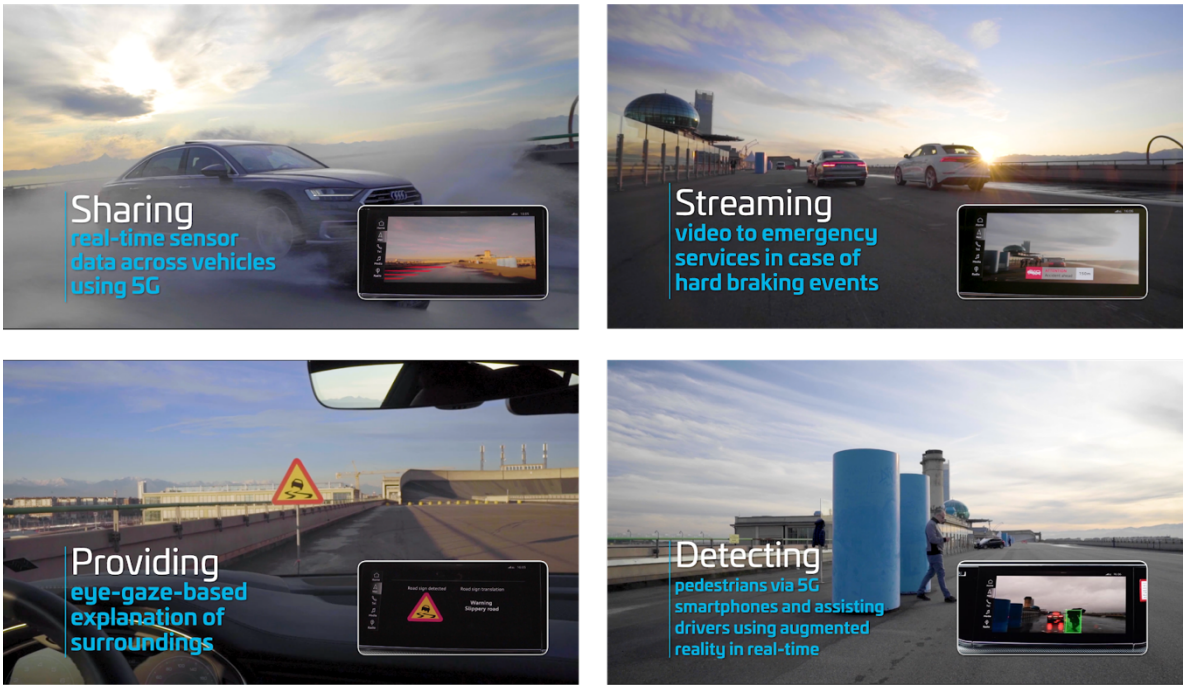



**Figure 47: Overview of the use cases as they were demonstrated during the event. The images in the video have been composed based on a video made by the 5GAA [55].**

## 10.2 Future Work

Although many different ideas eventually made their way into the final product that was shown during the conference, not all ideas were selected to be implemented. Given the logistics of the event - especially the hard requirement that the demo should not take longer than 5 minutes - some ideas couldn't possibly be squeezed into the available time. Two of these non-implemented ideas particularly stand out, because they would have enabled visitors to take a more active role within the

demo. The first idea, it was considered to have the visitors wear the Tobii Pro Glasses. One of the main roadblocks here was the calibration procedure. There was unfortunately not enough time to recalibrate for every participant in between the runs of the demo. Should a future demo or project aim to do something, then it would be advisable to use the same Tobii API interface that was used within this project. As part of that API, the calibration procedure can be triggered without the need to start the Tobii software, which is the most time-consuming procedure of the calibration. The other idea was initially planned to be implemented but was canceled due to the path visitors had to take after the end of our demo. In this idea, participants would be able to try out the app that was developed for the VRU use case, and see the warning message appear on the screen of the smartphone.

Besides reconsidering the suggestions mentioned above, there are also other possible challenges to explore in future work. For example, another possible angle for future work could be centered around how this setup could be made "transportable." Currently, two vehicles and many other pieces of side-equipment were necessary to demonstrate the use cases. Not only does this make the setup costly to replicate, it also prevents the demo from being shown at locations where this amount of required space is not available. Alternatively, at the cost of immersion, it could be explored if the demo could be shrunk to a simulator environment.

## 10.3 Conclusion

As part of this chapter of the thesis, four different ADAS use cases were implemented in an actual vehicle. The use cases spanned various themes, ranging from sensor sharing in a vehicle-to-vehicle scenario to a safety system for vulnerable road users through in a vehicle-to-anything setting. This work was the world's first ever demonstration of such a system using a 5G network to enhance its functionalities. To achieve this, various teams from different companies had to collaborate to make this happen. The work was successfully demonstrated at the 5G Automotive Association (5GAA) “The 5G Path of Vehicle-to-Everything Communication: From Local to Global” conference in Turin, Italy. During the day, more than 200 participants experienced the demo and the work was covered in various press releases.

## References


[1] '5G Localisation and Context-Awareness'. [Online] Available: https://www.5gitaly.eu/2018/wp-content/uploads/2019/01/5G-Italy-White-eBook-5G-Localization.pdf. [Accessed 22-Feb-2020].

[2] 'How 5G will impact the transportation industry'. [Online] Available: https://www.business.att.com/learn/tech-advice/how-5g-will-impact-the-transportation-industry.html [Accessed 22-Feb-2020]

[3] Taeihagh, A., & Lim, H. S. M. (2019). Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks. Transport reviews, 39(1), 103-128.

[4] 'Tesla Autopilot'. [Online] Available: https://www.tesla.com/autopilot. [Accessed 18-Feb-2020].

[5] 'Automated Vehicles for Safety'. [Online] Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety. [Accessed 18-Feb-2020].

[6] 'Advanced driver-assistance systems'. [Online]. Available: https://en.wikipedia.org/wiki/Advanced_driver-assistance_systems. [Accessed: 12-Jun-2019]

[7] 'US: Mobileye intros smartphone connected driver assistance (ADAS) technology'. [Online]. Available: https://web.archive.org/web/20120610055853/http://telematicsnews.info/2012/01/12/us-mobileye-intros-smartphone-connected-driver-assistance-adas-technology_j3122. [Accessed: 12-Jun-2019]

[8] 'What is 5g?'. [Online]. Available: https://www.ericsson.com/en/5g/what-is-5g. [Accessed: 12-Jun-2019]

[9] 'Everything you need to know about 5G. [Online]. Available https://spectrum.ieee.org/video/telecom/wireless/everything-you-need-to-know-about-5g. [Accessed: 12-Jun-2019]

[10] '5G networks will be 10 times faster than 4G LTE, but we shouldn't get too excited yet'. [Online]. Available: https://www.businessinsider.com/5g-high-speed-internet-cellular-network-issues-switch-2019-4?r=US&IR=T. [Accessed: 12-Jun-2019]

[11] 'F2F Meeting, Conference and 5GAA Demonstrations – Turin, Italy'. [Online]. Available: https://5gaa.org/calendar/f2f-meeting-conference-and-5gaa-demonstrations/. [Accessed: 22-Feb-2020]

[12] 'The Top Countries with 5G Deployments and Trials'. [Online]. Available: https://www.sdxcentral.com/5g/definitions/the-top-countries-with-5g-deployments-and-trials/. [Accessed: 22-Feb-2020]

[13] 'DRAFT REPORT on autonomous driving in European transport. [Online]. Available: https://www.europarl.europa.eu/doceo/document/TRAN-PR-623787_EN.pdf. [Accessed: 25-Feb-2020]

[14] 'Automated Vehicle Traffic Control Tower: Phase 2'. [Online]. Available: https://www.itrl.kth.se/research/ongoingprojects/automated-vehicle-traffic-control-tower-phase-2-1.917776. [Accessed: 22-Feb-2020]

[15] Ferguson, C., Davidson, P. M., Scott, P. J., Jackson, D., & Hickman, L. D. (2015). Augmented reality, virtual reality and gaming: an integral part of nursing.

[16] T Caudell and D Mizell, 'Augmented reality: an application of heads-up display technology to manual manufacturing processes', Proc. Twenty-Fifth Hawaii Int. Conf. Syst. Sci., vol. 2, pp. 659–669, 1992.

[17] 'Head-up display'. [Online]. Available: https://www.audi-technology-portal.de/en/electrics-electronics/controls/head-up-display_en. [Accessed: 22-Feb-2020]

[18] 'Embedded Holographic AR Display'. [Online]. Available: https://wayray.com/embedded. [Accessed: 22-Feb-2020].

[19] Holger Regenbrecht, Gregory Baratoff, and Wilhelm Wilke. Augmented reality projects in the automotive and aerospace industries. IEEE Computer Graphics and Applications, 25(6): 48–56, 2005.

[20] Jeamin Koo, Jungsuk Kwac, Wendy Ju, Martin Steinert, Larry Leifer, and Clifford Nass. Why did my car just do that? explaining semi-autonomous driving actions to improve driver understanding, trust, and performance. International Journal on Interactive Design and Manufacturing (IJIDeM), 9(4):269–275, 2015.

[21] Kun, A. L., Meulen, H. V. D., & Janssen, C. P. (2017). Calling while driving: An initial experiment with HoloLens.

[22] Tran, C., Bark, K., & Ng-Thow-Hing, V. (2013, October). A left-turn driving aid using projected oncoming vehicle paths with augmented reality. In Proceedings of the 5th international conference on automotive user interfaces and interactive vehicular applications (pp. 300-307).

[23] Rusch, M. L., Schall Jr, M. C., Gavin, P., Lee, J. D., Dawson, J. D., Vecera, S., & Rizzo, M. (2013). Directing driver attention with augmented reality cues. Transportation research part F: traffic psychology and behaviour, 16, 127-137.

[24] 'Object Detection' [Online]. Available: https://en.wikipedia.org/wiki/Object_detection. [Accessed: 22-Feb-2020].

[25] Zhao, Z. Q., Zheng, P., Xu, S. T., & Wu, X. (2019). Object detection with deep learning: A review. IEEE transactions on neural networks and learning systems, 30(11), 3212-3232.

[26] Redmon, Joseph (2016). "You only look once: Unified, real-time object detection". Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[27] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[28] 'How do Tobii Eye Trackers work?' [Online]. Available: https://www.tobiipro.com/learn-and-support/learn/eye-tracking-essentials/how-do-tobii-eye-trackers-work/. [Accessed: 22-Feb-2020].

[29] 'WebGazer.js Democratizing Webcam Eye Tracking on the Browser' [Online]. Available: https://webgazer.cs.brown.edu/. [Accessed: 10-Jul-2019].

[30] Gao, X. Y., Zhang, Y. F., Zheng, W. L., & Lu, B. L. (2015, April). Evaluating driving fatigue detection algorithms using eye tracking glasses. In 2015 7th International IEEE/EMBS Conference on Neural Engineering (NER) (pp. 767-770). IEEE.

[31] Hayami, T., Matsunaga, K., Shidoji, K., & Matsuki, Y. (2002, September). Detecting drowsiness while driving by measuring eye movement-a pilot study. In Proceedings. The IEEE 5th International Conference on Intelligent Transportation Systems (pp. 156-161). IEEE.

[32] Sodhi, M., Reimer, B., & Llamazares, I. (2002). Glance analysis of driver eye movements to evaluate distraction. Behavior Research Methods, Instruments, & Computers, 34(4), 529-538.

[33] Zheng, R., Nakano, K., Ishiko, H., Hagita, K., Kihira, M., & Yokozeki, T. (2015). Eye-gaze tracking analysis of driver behavior while interacting with navigation systems in an urban area. IEEE Transactions on Human-Machine Systems, 46(4), 546-556.

[34] Bastian Hinterleitner, Thomas Hammer, Stefan Mayer, Frederik Naujoks, and Nadja Schömig, 'Analyzing Gaze Behavior Prior to Interacting with a Multimedia Interface in a Car', Hum.-Comput. Interact. Interact. Technol. 20th Int. Conf. HCI Int. 2018 Las Vegas NV USA, vol. Part III, pp. 258–268, Jul. 2018.

[35] 'Augmented Reality' [Online]. Available: https://en.wikipedia.org/wiki/Augmented_reality. [Accessed: 25-Feb -2020].

[36] 'About openFrameworks' [Online]. Available: https://openframeworks.cc/about/. [Accessed: 10-Jun-2019].

[37] 'Tobii Pro Glasses 2' [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/. [Accessed: 10-Jun-2019].

[38] 'Tobii Pro Glasses 2 API' [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-glasses-2-sdk/. [Accessed: 10-Jun-2019].

[39] Davide De Tommaso and Agnieszka Wykowska. 2019. TobiiGlassesPySuite: An open-source suite for using the Tobii Pro Glasses 2 in eye-tracking studies. In 2019 Symposium on Eye Tracking Research and Applications (ETRA '19), June 25–28, 2019, Denver , CO, USA. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3314111.3319828

[40] Bradski, G. (2000). The OpenCV LibraryDr. Dobb's Journal of Software Tools.

[41] 'VideoCapture Class Reference' [Online]. Available: https://docs.opencv.org/3.4/d8/dfe/classcv_1_1VideoCapture.html. [Accessed: 10-Jun-2019].

[42] 'ofxUDPManager' [Online]. Available: https://openframeworks.cc/documentation/ofxNetwork/ofxUDPManager/. [Accessed: 10-Jun-2019].

[43] AlexeyAB, Yolo-V3 and Yolo-V2 for Windows and Linux, (2020), https://github.com/AlexeyAB/darknet/commits/master. [Accessed: 5-Mar-2020].

[44] 'Minimum bounding box' [Online]. Available: https://en.wikipedia.org/wiki/Minimum_bounding_box. [Accessed: 5-Mar-2020].

[45] 'Yolo: Real-Time Object Detection' [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed: 5-Mar-2020].

[46] 'Yolo for openFrameworks' [Online]. Available: https://github.com/MdMxMyr/Yolo_for_openFrameworks_Guide. [Accessed: 5-Mar-2020].

[47] 'Reynolds Implementation for openFrameworks' [Online]. Available: https://github.com/MdMxMyr/Yolo_for_openFrameworks_Guide/tree/master/Reynolds%20implementation. [Accessed: 5-Mar-2020].

[48] "Speeded up detection of squared fiducial markers", Francisco J.Romero-Ramirez, Rafael Muñoz-Salinas, Rafael Medina-Carnicer, Image and Vision Computing, vol 76, pages 38-47, year 2018

[49] "Generation of fiducial marker dictionaries using mixed integer linear programming",S. Garrido-Jurado, R. Muñoz Salinas, F.J. Madrid-Cuevas, R. Medina-Carnicer, Pattern Recognition:51, 481-491,2016

[50] Reynolds, C. W. (1999, March). Steering behaviors for autonomous characters. In Game developers conference (Vol. 1999, pp. 763-782).

[51] kylemcdonald, ofxCv, (2019), https://github.com/kylemcdonald/ofxCv. [Accessed: 13-Sept-2019].

[52] Lund, I. O., & Rundmo, T. (2009). Cross-cultural comparisons of traffic safety, risk perception, attitudes and behaviour. Safety Science, 47(4), 547-553.

[53] Hui, J. (2018). Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3. Retrieved from https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

[54] J Utah. (2018, 11 9). Boston 4K - Main Street - Driving Downtown - USA. YouTube. https://www.youtube.com/watch?v=XLh4Yt10wb4

[55] 5G Automotive Association. (2020, February). Turin Demo Wrap Up. Vimeo. https://vimeo.com/386944767

[56] Zorloni L. (2019, Nov). 'Il 5G al volante: ecco come renderà l'auto più sicura'. Wired. Available at: https://www.wired.it/internet/tlc/2019/11/16/5g-auto-sicurezza/

[57] 'C-V2X, Auto connesse 5G in Italia: cresce esponenzialmente la sicurezza'. (2019 Nov) Automoto. Available at: https://www.automoto.it/news/c-v2x-auto-connesse-5g-in-italia-cresce-esponenzialmente-la-sicurezza-gallery-video.html

[58] Borgomeo, V. (2019, Nov). 'Il cyber pneumatico è fra noi: così la gomma ti avvisa dei pericoli della strada'. La Repubblica. Available at: https://www.repubblica.it/motori/sezioni/sicurezza/2019/11/15/news/il_cyber_pneumatico_e_fra_noi_cosi_la_gomma_interagisce_con_il_5g-241166714/

[59] 'Pirelli lancia il primo pneumatico che dialoga con la rete 5G'. (2019 Nov). Ansa. Available at: http://www.ansa.it/canale_motori/notizie/componentie_tech/2019/11/14/pirelli-lancia-primo-pneumatico-che-dialoga-con-la-rete-5g_33db81c6-4e5c-4118-83d1-3dbf186c42fa.html

[60] Greco, F. (2019, Nov). 'Guida assistita, a Torino i test «on the road» di sei applicazioni'. Ilsole24Ore. Available at: https://www.ilsole24ore.com/art/guida-assistita-torino-test-on-the-road-sei-applicazioni-AC9Fvvy

[61] 'DA PIRELLI PNEUMATICI CHE INTERAGISCONO CON LA RETE 5G '. (2019 Nov). Il Tempo. Available at: https://www.iltempo.it/italpress/2019/11/14/news/da-pirelli-pneumatici-che-interagiscono-con-la-rete-5g-1240363/

# Appendix A: Yolo Installation Instructions for openFrameworks

**Note: this document can also be found on** https://github.com/MdMxMyr/Yolo_for_openFrameworks_Guide
**Where t has active links to the mentioned resources**

## Part one: installing Darknet and YoloV2

Please note, these instructions were written for a computer running the Windows 10 Operating System. Furthermore, it is recommended that the computer you're currently working with has an NVIDIA GPU installed. If not, you can try to follow the guide at your risk and skip all CUDA related steps in it.

First, there is a list of resources that need to be downloaded:

- This forked Darknet repository by AlexeyAB https://github.com/MdMxMyr/darknet
- Visual Studio 2017 Community Edition which can be downloaded at Visual Studio Older Downloads - 2017, 2015 & Previous Versions
- CMake >3.8 which can be downloaded from CMake Downloads
- The NVIDIA CUDA toolkit 10.0 CUDA Toolkit 10.0 Archive
- cuDNN Archive. Note, that you need the cuDNN for CUDA 10.0 for Windows 10 specifically
- OpenCV 3.4.6 for Windows, which can be downloaded from OpenCV Releases

Installation instructions:

1. First, and foremost, install Visual Studio 2017 (VS2017) before installing the NVIDIA toolkit. If you already installed CUDA before installing VS2017, it is recommended to uninstall CUDA and do a reboot first. During the installation process, the Wizard will ask if you want to install additional Workloads and Individual Components. Select the following Workloads:
    1. .NET desktop development
    2. Desktop development with C++
    3. Universal Windows Platform Development
    4. Python development

    Finally, make sure the following Individual Components (second tab) have been selected. Do note, not all of them are selected by default, so please check with care:
        5. C# and Visual Basic Roslyn compilers
        6. MSBuild
        7. Static analysis tools
        8. .NET Framework 4.6.1 SDK
        9. .NET Framework 4.6.1 targeting pack
        10. Text Template Transformation
        11. Visual Studio C++ core features
        12. Visual C++ 2017 Redistributable Update
        13. VC++ 2017 version 15.9 v14.16
        14. Windows 10 SDK (10.0.17763.0)
        15. Windows Universal CRT SDK
        16. Windows 8.1 SDK
        17. C++/CLI support
        18. VC++ 2015.3 v14.00 (v140) toolset for desktop
        19. Visual C++ compilers and libraries for ARM64
        20. C++ Universal Windows Platform tools for ARM64
        21. Visual C++ compilers and libraries for ARM
        22. Visual C++ tools for CMake and Linux

23. VC++ 2017 version 15.9 v14.16 Libs for Spectre (ARM)
24. VC++ 2017 version 15.9 v14.16 Libs for Spectre (ARM64)
25. VC++ 2017 version 15.9 v14.16 Libs for Spectre (x86 and ARM64)

3. After having selected all the Workloads and Individual Packages, you can download and install them. This process will take a while, providing an excellent moment to grab a coffee while the installer does its work. After the installation is complete, make sure to run VS2017 at least once and reboot your computer before doing the next step.

4. In case you haven't done so already, it is worth to make sure your GPU drivers are up to date. Given that you're following this guide, it is safe to assume that the computer you're currently working with has an NVIDIA GPU. Probably the easiest way to keep your GPU drivers up to date is through NVIDIA GeForce Experience which can be downloaded from NVIDIA's GeForce Experience website. Make sure you also check if your GPU has the latest drivers after installing

5. Once your system is rebooted, it is time to install CUDA 10.0. This process is very straightforward; just follow the standard installation procedure given by the installation Wizard. Note that during the end it will notify you that CUDA tools for VS2017 has also been installed (hence why it needs to be installed prior to CUDA). Once again, reboot your computer so that the Windows PATH variables can be updated.

6. After the reboot, the cuDLL libraries can be added to CUDA. Start by extracting the ZIP file that was downloaded earlier. Then, add the contents of the zip file to the directories in the CUDA installation folder. By default, the directory path is `C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0` (now referred to as CUDA_PATH):
    1. Add 'cudnn64_7.dll' to `CUDA_PATH/bin`
    2. Add 'cudnn.h' to `CUDA_PATH/include`
    3. Add 'cudnn.lib' to `CUDA_PATH/lib/x64`

7. Then, it is time to extract OpenCV. Open the OpenCV executable downloaded earlier and choose a directory where you would like to install OpenCV. I would personally recommend something like `C:\opencv\3.4.6`

8. Before it is time to build the repository, it is advised to check if the System Variables and PATH variables have been set accordingly. In case you have rebooted your system during the earlier steps, most of the process should have happened automatically already, but checking doesn't you any extra.
   For the System Variables, check if these exist, and if not create them. In case you had to add one, make sure to reboot your system after finishing this step.
    1. Make sure there is a "CUDA_PATH" and a "CUDA_PATH_V10_0" variable that have an address assigned to it which is identical to the address used step 4.
    2. Also check for a "OpenCV_DIR" System Variable with an address linking to `OPENCV_DIR\build\x64\vc15` where OPENCV_DIR is the directory at which you've extracted it in step 5.
    3. Double click on PATH in System variables and make sure the following paths are present: `CUDA_PATH\bin` and `CUDA_PATH\libnvvp` where CUDA_PATH is the directory used in step 4.

9. Create a new directory where you would like to unpack Darknet. Then, extract and add the Darknet Repository ZIP file in this directory. Remember this location for step 9. This location will now be referred to as `DARKNET_PATH`.

10. Now, extract the CMake ZIP, and open cmake-gui in the /bin directory:
    1. Once the GUI appears, click on “Browse Source” and add the `DARKNET_PATH/darknet-master`
    2. Then, click on “Browse Build” and add the same directory; `DARKNET_PATH/darknet-master`
    3. Click on Configure and make sure “Visual Studio 15 2017 win64” is selected as the generator for this project. Then press configure again.
    4. Now, an error may appear stating that a configuration openCVConfig.cmake has not been found. You can add this one manually by clicking on “Ungrouped Entities” and then on OpenCV_DIR and replace the value with the following path: `C:\opencv\3.4.6\opencv\build\x64\vc15\lib` (or your own OPENCV_PATH). Then hit the configure button again

11. Now press Generate, and CMake should state “Generating done”. You can now close CMake.

12. Go to `DARKNET_PATH/darknet-master/build/darknet` and open yolo_cpp_dll.sln. In case VS2017 asks if you’d like to upgrade the solution, do not do this. Make sure you’ve set the compiler to “Release” and “x64”. Then, build the solution in the solution Explorer Menu. Should the compiler throw an error that “opencv” or “opencv2” couldn't be found, then make sure `C:\opencv\3.4.6\opencv\build\include` is included as an Additional Include Directory. You can navigate to this menu through the following path: `Project > Properties > Configuration Properties > C/C++ > General > Additional Include Directories`. Then, you need to go to `Project > Properties > Configuration Properties > Linker > General > Additional Library Directories` and add `C:\opencv\3.4.6\opencv\build\x64\vc15\lib`

13. Now open `yolo_console_dll.sln`. Again, do not update in case prompted. Make sure you’ve set the compiler to “Release” and “x64”. Build this solution as well.

14. Finally, download a set of weights for Yolo V3 from the authors Dropbox (weights stored for archiving purposes, but obtained from PJR Eddy originally) and add these to `DARKNET_PATH\darknet-master\build\darknet\x64`. Now, if you run `darknet_yolo_v3.exe`, a console window should open and after a while, a YOLO-classified image of a dog and a bike should appear.

15. After all these steps, you’ve just installed Yolo V3, congratulations!

### Part 2 - Including YOLO V3 in openFrameworks

Now that you have YOLO V3 up and running, you might want to include it in an OpenFrameworks project. This section aims to give you a quickstart to achieve this. This guide assumes that you’ve followed the steps in the previous section.

1. First, you will need to download the openFrameworks (OF) library for VS2017. You can download the library from their website.
2. Although it is not necessary, it is recommended to also install the OF addon for VS2017. You can do this through launching VS, then in the toolbar, go to `Tools >`

`Extensions and Updates…`. From there, search for "openFrameworks plugin for Visual Studio 2017" by Arturo Castro and Half Scheidl. Install it and reboot VS.

3. Once VS is rebooted, in the toolbar, go to `File > New > Project…` and then navigate to `Visual C++ > openFrameworks` to create a OF-project. The project-Wizard will then ask you for the path to your OF-library. This is the Zip file you downloaded in step 1 of this section. Unpack the library and set the path to the library-directory. The Wizard will test if you've linked the right directory in case of doubt.
4. Once your project is generated, it is recommended to test if everything works thus far by trying to build the "starter project". You can do this by pressing CTRL + F5 in VS. Make sure your compiler is again set to "Release" and "X64". If, after compiling, you're presented with both a Command Prompt and another window with an empty-grey background, all works as intended.
5. Now, comes the tricky-part, which is adding the YOLO V2 library (note, V3 uses the V2 library for some reason) to the project:
    1. First, in the Solution Explorer, right-click on your projects name (note, not the solution itself nor the OF library) and go to `Add > Existing Item…`. From the menu, you want to go to the Darknet Build directory, which is `DARKNET_PATH/darknet-master/build/darknet/x64` and select the "yolo_cpp_dll.dll" and "yolo_cpp_dll.lib".
    2. Then, you should add OpenCV to this project as well. This goes similarly as in the previous guide when you build YOLO V3 library: in VS2017, go to `Project > Properties > Configuration Properties > C/C++ > General > Additional Include Directories` and add `C:\opencv\3.4.6\opencv\build\include` is included as an Additional Include Directory. Make sure that you Apply the changes when closing the window. Finally, you need to go to `Project > Properties > Configuration Properties > Linker > General > Additional Library Directories` and add `C:\opencv\3.4.6\opencv\build\x64\vc15\lib`. Once again, remember to click on Apply.

6. Now, you should be able to import YOLO V3 into your project in openFrameworks. The project should compile without issues, but it doesn't execute correctly, as some critical resources are missing. Therefore, you will still need to add some files to the `bin` directory of your project. Click on the project in the Solution Explorer and go to "Open Folder in File Explorer" and go to the bin-directory. Now to this directory, copy the following files:
    1. From `DARKNET_PATH/darknet-master/build/darknet/x64` copy "yolo_cpp_dll.dll". Also, copy the "yolov3.weights" that were downloaded and copied to this version in the previous section. (Link to download the weights)
    2. Also copy the entire `DARKNET_PATH/darknet-master/build/darknet/x64/cfg` directory, as well as the `DARKNET_PATH/darknet-master/build/darknet/x64/data` directory.
    3. From `DARKNET_PATH/darknet-master/3rdparty/pthreads/bin` copy pthreadVC2.dll

7. After having completed these steps, go to this Github Repository and copy the code from `ofApp.cpp` and `ofApp.h` to your own project (you can also replace the files in the `/src` directory of your OF project). Now when you compile the project, the code shall output the detected objects to the Command-Prompt window. If this all works, you've got YOLO V3 in openFrameworks all set up!

# Appendix B: Dangerous events marked by participants

This appendix contains the analysis of eleven events of which most participants, across each testing-cycle, marked them as dangerous.

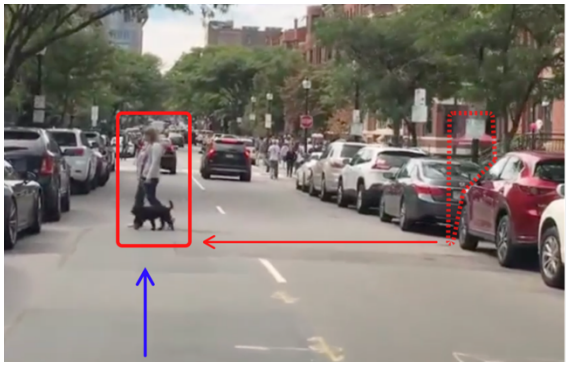
In the first section, annotations have been provided to further context, as well as which group of participants recognized the dangers first. For each event, the person (or group) that caused the dangerous event is marked with a red border for visibility. To illustrate their path, arrows have been added to the images. Additionally, an outline has been added to demonstrate where the cause-of-danger was initially coming from. A blue indicator has been added to illustrate the path of the vehicle.

The second section contains an overview for each event showcasing the exact order in which each group managed to spot the danger first. To determine this, the average performance of the group was considered.

## Section 1

### 1. Jaywalker with dog; From the far-away

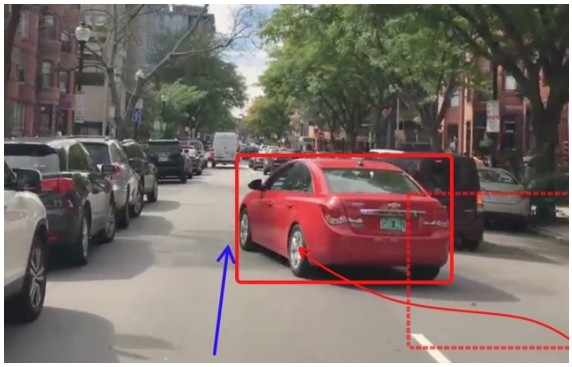
**The event**
A pedestrian happens to cross the street while walking their dog. The vehicle needs to slow down to let them pass safely.



**Observation**
What stood out from this event, is that the group that did not receive any augmentations, didn't seem to recognize the danger. It can therefore be assumed that they were unable to recognize the pedestrian well enough without any augmentations applied to the video. Once the augmentations were applied, the participants were able to recognize the event.

### 2. Red car switching lanes; From the right lane

**The event**
The red vehicle overtakes the vehicle in front of it (it's standing still). It does so, without giving any sign with their blinkers. The vehicle needs to slow down to let the vehicle pass.



**Observation**
The participants that were shown this video in a lower video quality without any augmentations performed the worst. The participants that saw the video in a high quality performed similarly to the groups that saw the video in lower quality with augmentations applied to it (for both smart and basic augmentations).

### 3. Van-driver leaving the vehicle: From the far-left

**The event**

While the vehicle is waiting at the intersection, there is a dry-cleaning-service van on the left from which a person appears through it's side-door. The danger lies in the fact that the man might leave the vehicle and stand in front of the vehicle.

**Observation**

What makes this case interesting compared to many others, is that no augmentations are being applied to the danger in this situation. The augmentations were only applied to pedestrians and vehicles. The clothes were not recognized as part of it. However, what is noticeable in the data, is that the participants that saw this video in lower quality with smart augmentations applied to it, didn't seem to notice the danger at all (with the exception of one participant, but much later than all other participants). A possible explanation is that this event happened away from the center of the screen and that this area didn't get a lot of eye-gazes during the tests with the smart augmentations being applied.

### 4. Jaywalker appearing behind a vehicle: From near the centre

**The event**

As the vehicle is overtaking a vehicle that is standing still, a pedestrian that has already started to cross the road pop-ups from behind the vehicle; the driver needs to slow down.

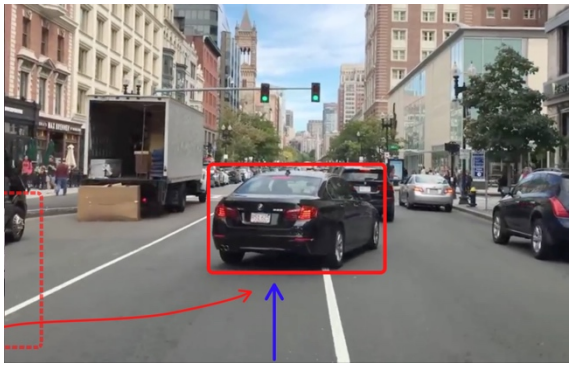
**Observation**

What stood out from this event is that the participants that did not receive any augmentations, while seeing this event in a low-quality video, did not recognize this danger at all. Additionally, in this case, the group with the low-quality video that did receive augmentations happened to press way earlier then the group that viewed in high video quality with no augmentations applied to it.

### 5. Vehicle changing lanes; Coming from the far-left lane

**The event**

As the vehicle is approaching the traffic lights, a vehicle from two-lanes to the left suddenly changes lanes to the rightest lane.

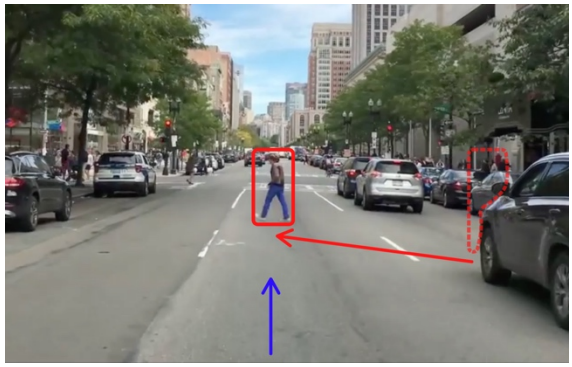


**Observation**

In this event, the group that saw this in low quality with smart augmentations performed just as well as the group that saw this video in high quality. This event is, therefore, an exception where this group outperformed the others, even with the danger coming from the side. The most likely explanation for this might be that the smart augmentations were also applied for objects moving to the center of the screen in a fast way. Therefore, this vehicle also got a border applied to it, even though it can all the way from the side. Like most events, the group that saw this event in low video quality without augmentations performed the worst, recognizing the danger much later.

## 6. Jaywalker crossing the street; Coming from the right

### The event
A pedestrian happens to cross the street from the right. The vehicle does not need to slow down just yet though, as it happens from quite far away.



### Observation
Although, strictly speaking, there is no danger yet, the participants still marked this as a dangerous scenario. What is interesting about this event, regardless, is that the participants who saw this event in low quality with smart augmentations waited much longer before marking this event. Given the distance at which the pedestrian crossed the road, the border color that was applied was subtle and green at the start of the event. As the color switches to yellow, participants started to mark the event as dangerous. It is interesting to consider what potential influence the color-shift of the augmentation-border possibly had on the assessment of the level of danger.

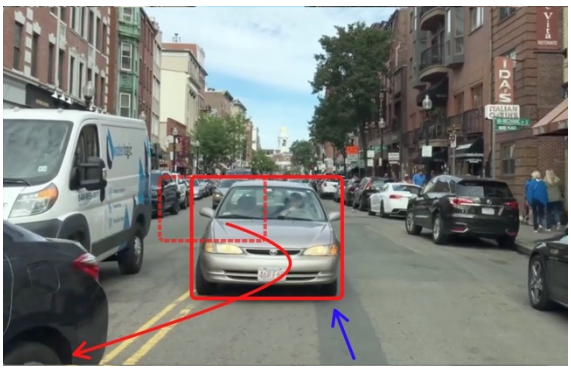
## 7. Vehicle overtaking another vehicle; From near the centre

### The event
A vehicle decides to overtake a van that's standing still. Multiple vehicles follow this vehicle. In this situation, the focus is on the first vehicle that makes this maneuver. The driver needs to slow down because of this.

### Observation
Like other events where the event starts from near the center of the screen, the participants that see this video in low quality with smart augmentations outperform the others. By just a small margin, however, the participants that saw this event in high quality outperformed them.

## 8. Vehicle overtaking another vehicle; From near the centre

### The event
This event happens just after the events of 7. Multiple vehicles follow the first car. By now however, the driver needs to stop completely in order to also let this last vehicle through.



### Observation
Like the previous event, where the event starts from near the center of the screen, the participants that see this video in low quality with smart augmentations outperform the others.

## 9. Group of jaywalkers: From the right near the centre

### The event
A group of jaywalkers decides to cross the street. The vehicle needs to stop completely to let the stream of pedestrians pass.

### Observation
The participants that had augmentations applied to their low-quality video performed like the participants that saw this event in high quality. The overall observation is very similar to event 6. However, this time the event happened closer to the center of the screen: which is what gave the smart augmentation a slight edge here.

## 10. Vehicle overtaking another vehicle; From near the centre

### The event
From the far left, a pedestrian decides to run across the street. The vehicle needs to slow down as a result of this.

### Observation
The participants that saw this event in a low quality with augmentations, outperformed the participant seeing this video in high quality. The basic augmentations performed best here. The most plausible explanation for this would be that for the smart augmentations no border would be shown initially (given that the pedestrians comes from the far left), and that these participants mostly focused around the center of the screen; away from the pedestrian.

## 11. Vehicle overtaking another vehicle; From near the centre

### The event
A vehicle decides to overtake a van that's standing still. The driver needs to slow down as the car partly blocks the other lane during this maneuver.

### Observation
Again, for events that happen near the center of the screen, the participants that witness the event in low quality with smart augmentation are the first to notice it. This is then followed by the high quality video participants.

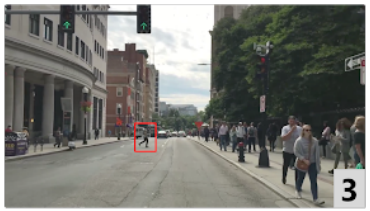
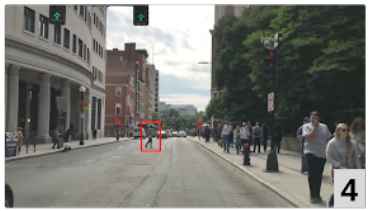
# Section 2


Subject who causes the event;
From where they enter the view
For each testing-cycle;
Ranking on recognizing the danger (averaged)
1 Jaywalker with dog:
From the right from far-away
2 Red car switching lanes;
From the right lane
3 Van-driver leaving the vehicle:
From the far-left
4 Jaywalker appearing behind a vehicle:
From near the centre
No Augmentations -
High Video Quality
No Augmentations -
Low Video Quality
Basic Augmentations -
Low Video Quality
Smart Augmentations -
Low Video Quality

**Subject who causes the event;**
**From where they enter the view**

**For each testing-cycle;**
**Ranking on recognizing the danger (averaged)**

| | No Augmentations - High Video Quality | No Augmentations - Low Video Quality | Basic Augmentations - Low Video Quality | Smart Augmentations - Low Video Quality |
|---|---|---|---|---|
| **5** Vehicle changing lanes; Coming from the far-left lane |  |  1 |  3 |  2 |  1 |
| **6** Jaywalker crossing the street; Coming from the right |  |  1 |  2 |  3 |  4 |
| **7** Vehicle overtaking another vehicle; From near the centre |  |  1 |  2 |  2 |  1 |
| **8** Vehicle overtaking another vehicle; From near the centre |  |  2 |  3 |  2 |  1 |

| Subject who causes the event; From where they enter the view | For each testing-cycle; Ranking on recognizing the danger (averaged) | | | |
|---|---|---|---|---|
| | No Augmentations - High Video Quality | No Augmentations - Low Video Quality | Basic Augmentations - Low Video Quality | Smart Augmentations - Low Video Quality |
| **9** Group of jaywalkers: From the right near the centre |  |   |   |   |   |
| **10** Jaywalker runnnig across the street; From the far-left |  |   |   |   |   |
| **11** Vehicle overtaking another vehicle; From near the centre |  |   |   |   |   |

# Ranking overview

The table below provides an overview of how many times a certain event was spotted first, second, third, fourth, or not all, for each category.

What is noticeable, is that if that *one case* in which danger wasn't noticed by the smart augmentations, then it's score would have been similar to the high video quality's performance (based on this ranking model).

| | High Video Quality - No Augmentations | Low Video Quality - No Augmentations | Low Video Quality - Basic Augmentations | Low Video Quality - Smart Augmentations |
|---|---|---|---|---|
| **First to notice (1x)** | 8 | | 1 | 5 |
| **Second to notice (2x)** | 1 | 3 | 5 | 4 |
| **Third to notice (3x)** | 2 | 4 | 5 | |
| **Fourth to notice (4x)** | | 2 | | 1 |
| **Didn't notice at all (5x)** | | 2 | | 1 |
| **"Ranking"** | 16 | 36 | 26 | 21 |

TRITA-EECS-EX-2019:XX