Aalborg University

Medialogy, MTA15641

6th Semester - BSc Project: Interactive Systems Design

An Investigation in Traffic Sign Detection and Feedback in Advanced Driver Assistance Systems

Machine-learning and Integral Channel Features, with an evaluation on modality use in Advanced Driver Assistance Systems

May 27, 2015

AALBORG UNIVERSITY
STUDENT REPORT

AALBORG UNIVERSITY
STUDENT REPORT

Department for Architecture, Design, and Media Technology
Rendsburggade 14
DK-9000 Aalborg
+45 9940 8793
http://create.aau.dk

Theme: BSc Project:
Interactive Systems Design

Title: An Investigation in Traffic Sign Detection and Feedback in Advanced Driver Assistance Systems

Semester: 6th semester, Spring 2015

Members:

Christoffer Bøgelund Rasmussen

Joachim Egsgaard Pedersen

Lars Myrhøj Nielsen

Thomas Storm Pedersen

Supervisor: Andreas Møgelmose

Copies: 6

Pages: 97

Finished: 27-05-2015

Synopsis:

This report is based on the 6^{th} semester theme of Interactive Systems Design. In this project an initial analysis of traffic accidents and driver psychology was conducted. It was discovered that drivers can be distracted, inattentive, and selectively attend to stimuli. Therefore they may overlook critical information. Through state of the art research it was determined to look into the building blocks of an Advanced Driver Assitance System, where the aim was to detect speed signs and relay potential warnings unobtrusively. A computer vision, machine-learning approach was used for detection. Firstly, using the Integral Channel Features method, features for speed signs were calculated, these were then boosted towards an optimal strong rule, using AdaBoost. By using a Soft Cascade approach, it was possible to detect a speed sign in an image. The method lead to satisfactory detection rates, however, precision was poor. An experiment was conducted to evaluate the use of modalities in an ADAS; namely visual, audio, and haptic feedback. This was done in terms of intuitiveness, reaction time to warnings, cognitive load, and general driving behaviour. The experiment indicated strengths and weaknesses within each modality. The results indicated that an ADAS system should make use of multimodal design.

Preface

This report was created as a 6^{th} semester Medialogy bachelor project, with the theme Interactive Systems Designs. This project includes aspects of the two courses taught throughout the semester: Real-time Interfaces and Interactions, and Media Sociology and Psychology. The project revolves around of creating a system that is able to detect road signalisation and delivering potential warnings to the driver. This is done with a computer vision method called Integral Channel Features.

The project was carried out in the period of February 2015 to May 2015 at facilities for media technology at Aalborg University.

This bachelor project is created upon the knowledge gathered throughout the entire bachelor-programme for Medialogy. The knowledge gained in 3^{rd} semester, around image processing, was of large inspiration to the group

A DVD is enclosed containing the C++ project for training and detection, the trained classifier, and MATLAB scripts which were used for detector evaluation. Also included are data from the experiment and an A/V production, giving a visual presentation of the project.

A special thanks goes to our supervisor Andreas Møgelmose for his dedication and guidance throughout the process of this bachelor project. Especially his contribution of knowledge in regards to computer vision.

Contents

Chapter 1

Introduction

The theme of this semester is Interactive Systems Design. The purpose for the semester is to implement an interactive system, using knowledge gained throughout the Medialogy bachelor. This project aims to create a platform for an Advanced Driver Assistance System (ADAS) for traffic safety.

Traffic accidents account for millions of accidents every year across the globe. World Health Organization reported that traffic collisions account for 20-50 million injuries worldwide, and 1.2 million fatal accidents each year [World Health Organisation, 2004]. On a national scale, a total of 2,954 have died and another 51,063 have been injured as a result of traffic accidents in Denmark (2004-2013). During this period, however, the number of fatal accidents has dropped by almost 54%, as reported by the Danish ministry of traffic [Vejdirektoratet, 2014]. The reason behind this rapid decrease can be due to various developments, that all aim to aid the driver and increase traffic safety. Examples of such developments could be infrastructural changes, increased driver awareness, and passive- and active safety technologies. [Tran et al., 2012][Kovačić et al., 2013]

According to Tran et al. [2012], 80% of crashes and 65% of near crashes are direct results of human errors, such as inattention or cognitive overload. This is one of the motivations for developing and implementing ADAS. ADAS are examples of active safety technologies which have become more prevalent among car manufacturers in recent years. ADAS can be applied with various purposes, however, they all aim to increase safety and assist the driver, by predicting and potentially helping avoid accidents.

Tran et al. [2012] also state that in order for ADAS to be effective, it needs to be human-centric and take different components into account, such as the driver, the driving environment, and the ego vehicle. This data can be obtained using various technologies. Kovačić et al. [2013] present different approaches and solutions to the most common traffic accidents using computer vision. They argue that due to improvements in cheaper cameras and increasing computational power, computer vision methods already play a major role in reducing traffic accidents.

Based on the introductory research presented in this section, an initial problem statement

can be formed:

> *How can Advanced Driver Assistance Systems be used to reduce the number of traffic accidents?*

In order to properly investigate this initial problem statement, different areas must be investigated, such as magnitude of the problem, and psychology of driving. This will be presented in the following chapter.

Chapter 2

Problem Analysis

This chapter will present an in-depth investigation of the major aspects of the initial problem statement. First, the reasons behind traffic accidents are presented on a statistical background. Second, different psychological aspects of driving are investigated. Various existing computer vision approaches is also presented, as well how to convey information to the driver. Finally, a field study is conducted to get a first-hand impression of this research.

2.1 Magnitude of the Problem

As previously mentioned, 2,954 people have died due to traffic accidents, while 51,063 also occured in Denmark (2004-2013). While the number of fatal accidents has fallen by 54% and number of injuries by 60%, however, these can still be reduced even further. In 2011, The Danish Traffic Safety Commission (DTSC) set an objective to reduce both the number of people injured and killed in traffic accidents by 50% by 2020. This goal has been determined to be realistic if the necessary political decisions are made, and sufficient resources are made accessible. The goal also builds on a foundation of knowledge acquired through recent initiatives. DTSC [2013] also states that several other factors, such as driver education, road maintenance, and technological advancements in vehicles, such as ADAS, will help towards achieving the goal [DTSC, 2013]. A figure of the overall number of deaths and injuries on Danish roads over the past decade can be seen in Figure 2.1.

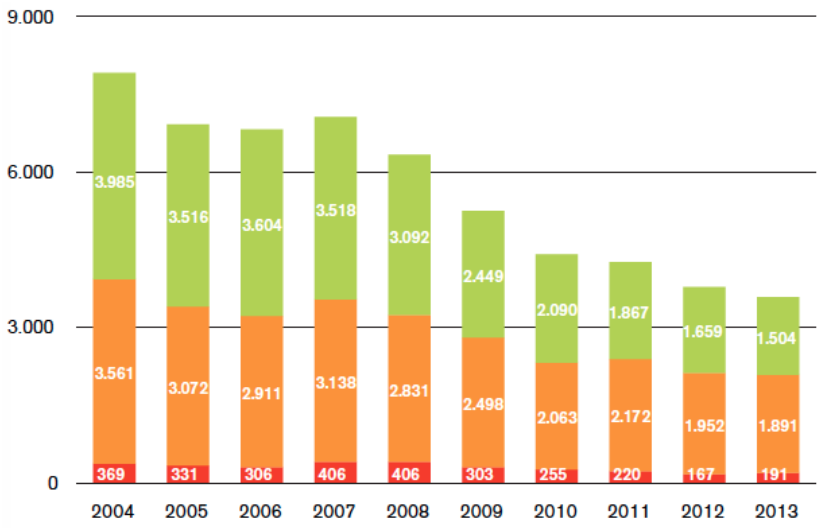


Figure 2.1: Number of traffic accidents between 2004-2013. Each year is split into the number of deaths (red), serious injuries (orange), and minor injuries (green) [Vejdirektoratet, 2014].

The majority of accidents occur in personal cars, where in 2013 there were 157 deaths and 2,864 accidents. After this, other types of vehicles are a roughly somewhat even distribution of deaths and accidents. This can be seen in Figure 2.2.

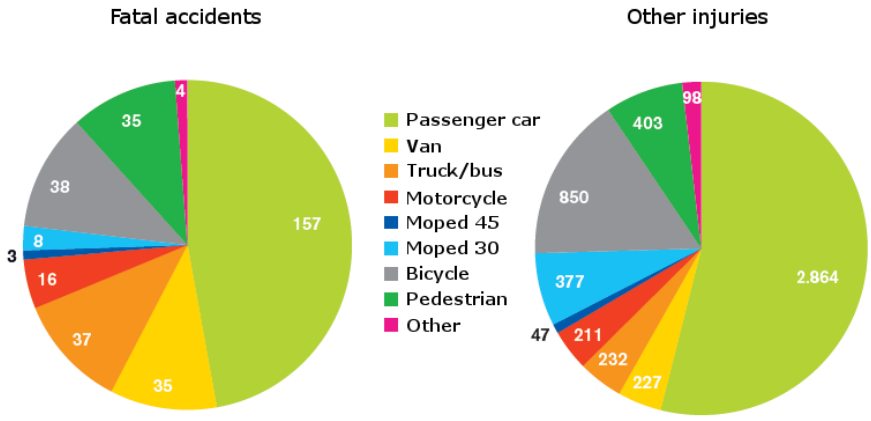


Figure 2.2: Spread of traffic deaths and accidents according to type of vehicle in Denmark 2013. [Vejdirektoratet, 2014]

Vejdirektoratet [2014], has categorised the types of accidents into nine different types. The top three categories in terms of both traffic deaths and accidents are solo accidents, head-on accidents, and accidents involving pedestrians. The different categories and the percentages of accidents and deaths within those categories can be seen in Figure 2.3.

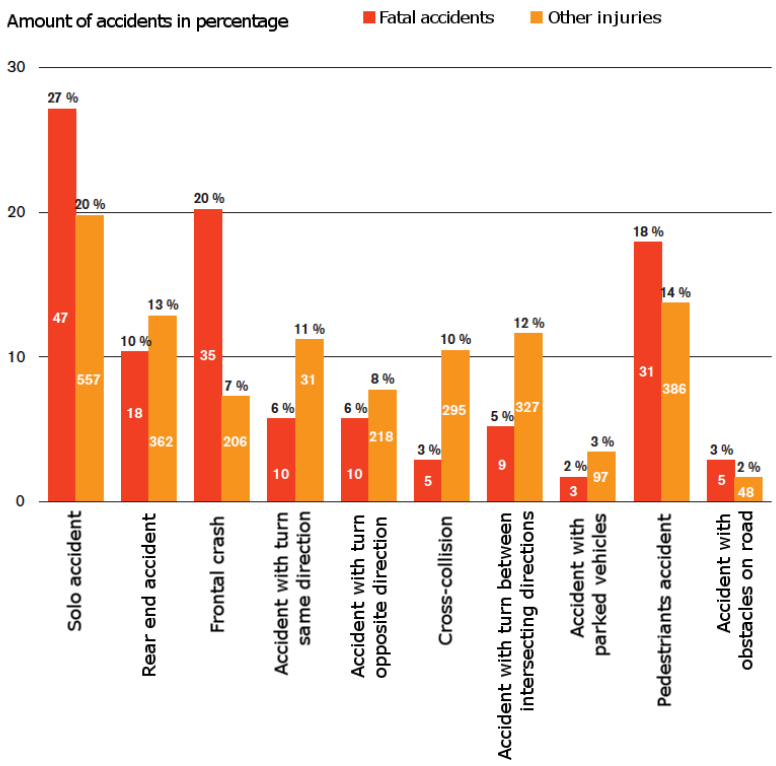


Figure 2.3: Spread of traffic deaths according to type of accident [Vejdirektoratet, 2014].

The report from Vejdirektoratet [2014] shows the factors behind individual accidents. An accident, however, might be a result of several different factors. For example, an accident might be caused by speeding after missing a speed sign, which in turn might be a result of inattentiveness or lack of orientation. Table 2.1 shows the four major factors leading to traffic deaths.

Factor	**Deaths**
Lack of and/or missing orientation	54
Influenced by alcohol/drugs/medicine	48
Lack of and/or missing attentiveness	41
Driver over the speed limit	40

Table 2.1: Overview of the main factors leading to traffic deaths in 2013. An accident might be caused by multiple factors [Vejdirektoratet, 2014].

In the following sections, different psychological aspects of driving will be investigated to gain a better understanding of the reasons behind these accidents.

2.2 Driver Information Processing

In general, driving is a fairly automated task for most people. While there are a lot of stimuli that must be processed and actions made accordingly, most drivers do this automatically. Driving does not always require 100% attention and many drivers think and do many other things along their journey. While this is fine in most cases, potentially deadly issues arise when something unexpected occurs and the brain does not have enough time to react to this situation. [Shinar, 2007b]

Driving involves many different forms of processing which are both conscious and subconscious, automated and controlled, expectations of traffic and road conditions. When driving there is a lot of stimuli that the driver encounters, so much that it is impossible for them to process it all completely. Shinar [2007b] explains how the brain decides to filter the stimuli that it receives, through four stages:

1. **Selective Attention:** In this stage the brain decides what to attend to. Much of this is based on external cues, such as bright colours on a sign and the expectations the driver has of the road, such as a stop sign when approaching an intersection.
2. **Information Value:** In the next stage the brain makes decisions on what the meaning of the selected stimuli is. Simply seeing the lights on the traffic lights is not beneficial, it is important to interpret what the different colours mean.
3. **Reacting to the Information:** Based on the external information and individual driving style, the brain decides on how to react to the stimuli.
4. **Overt Control Action:** After the first three stages, the driver performs an action that affects the vehicle in some way.

This process of reacting to stimuli is especially important in terms of safety. Shinar [2007b] states the two most common actions in this case as: steering and braking to avoid an accident. One major factor affecting these two actions is the perception reaction time. Many crashes can also be associated to a delayed recognition of danger to the driver. This can either be in terms of the critical event leading a crash not being recognised, or the reaction time was delayed so the driver did not have enough time to react.

Research has been conducted to see if there is a relationship between the visual fixations and perception the driver does, and recollection of objects in the driving scene. Here it was found that not all types of driving information is fixated on or recalled equally, instead, the driver weighted the type of information they deemed important. Here drivers fixated and recalled all of the speed signs, but a much smaller number fixated on a pedestrian crossing sign (21%) and none recalled the sign [Shinar, 2007b]. The results from the experiment can be seen in Table 2.2.

	Fixated		Not Fixated	
Target	Recalled	Not recalled	Recalled	Not recalled
Speed limit 80 km sign	100	0	0	0
Game crossing sign	60	0	7	33
Lane marking for right turn	93	7	0	0
Lane marking for left turn	7	0	0	93
No seperate lanes	38	8	54	0
Intersection w/o pedestrian crossing	47	7	33	13
Pedestrian crossing ahead	8	54	0	38
Pedestrian crossing sign	0	21	0	79
Crosswalk lines	29	50	7	14
Roadside billboards	20	23	0	57
Houses along the street	0	0	0	100

Table 2.2: Percentage of fixation and recollection rates of various objects immediately after the drivers passed them.[Shinar, 2007b]

Luoma [1991] conducted the experiment again and found similar results in terms of fixation and recollection rates. However, in this instance they attempted to see if drivers could react to stimuli without necessarily recalling it. Luoma [1991] found that different levels of processing were possible, with some drivers fixating on a sign, reacting accordingly but not recalling it. Others fixated on a sign, did not react accordingly but did recall it. The most notable piece of the experiment, in both studies, was that if the object was not fixated upon it was almost never recalled. It was found that while a driver is not actively searching for a sign the chance of them seeing it is based upon how visually prominent it is in the scene. This factors include the size, colour, and contrast to the scene.

The processing can also be dependent on how relevant it is to the drivers current task. For example when approaching an intersection, stop signs have a high level of fixation from the driver. While signs that are not directly relevant are not noticed as much. If a driver does not feel like they need to information it has a higher chance to be ignored.

2.3 Driver Attention

As mentioned in Section 2.1, distraction and inattention are two of the leading factors in accidents. Various studies have concluded that if drivers allocated all avaliable attentional capacities to driving, between 20-80% of accidents could be eliminated [Shinar, 2007a]. Shinar [2007a] lists three types of such factors:

1. **Inattention:** This is when the source of distraction is internal to the driver, such as daydreaming and general non-driving thoughts.
2. **Internal Distractions:** These are objects or events that occur inside the car, such as passengers, radio, etc.
3. **External Distractions:** Objects or events that occur outside of the vehicle, such as billboards, pedestrians, etc.

Inattention often happens as the task of driving can be quite monotonous, and drivers have a certain expectations while driving. As a result, the brain of the driver will seek to

occupy itself with non-driving tasks. However, completely removing inattention and making drivers concentrate on all stimuli that is relevant can be an exhausting task while very effective. A study was conducted where drivers were asked to do this, here they were able to recall 98% of all traffic signs. In comparison to other studies of normal driving conditions where drivers could only recall between 5-50%. [Shinar, 2007a]

Stutts et al. [2003] conducted an experiment where a large number of drivers were filmed, where 3 hours of video were gathered from each driver. It was found that drivers spent a large amount of time being distracted, with not a single person allocating all of their attention to driving. At some point 75% of drivers were either eating and/or drinking, controlling the radio, speaking with passengers, or adjusting the controls inside the car. In relation to the factors presented by Shinar [2007a], these are examples of internal distracions. Other distractions were also present, however, these were the most frequent.

Another study was conducted in Australia, where 1,367 drivers were interviewed after being involved in an accident. 32% admitted to being distracted while driving, and 14% considered it to be the case of the accident. However, Bunn et al. [2005] found that there was a link between the severity of crashes and distraction. They concluded that distracted or inattentive drivers were 3.2 times more likely to be involved in a fatal crash, compared to being only injured.

This leads back to the underlying theme that driving is a relatively easy task and many drivers have expectations of road situations and conditions that could lead to inattention and distractions. These issues once again mean that the driver will have a slow perception reaction time when presented with an unexpected situation, potentially leading to a fatal accident.

2.4 Reducing Traffic Accidents

As mentioned earlier, one method of reducing traffic accidents is by using ADAS and other technological advancements. An example of this is computer vision. This section will present some of the most common traffic accident causes, as well as different methods to help prevent these types of accidents using computer vision. Kovačić et al. [2013] gives examples of computer vision solutions to four different types of accidents. The following four subsections will present each of the four causes more elaborately, and discuss different computer vision approaches and solutions.

2.4.1 Frontal Crashes

Even though various types of systems exist which can help reduce the number of frontal vehicle crashes, computer vision is a great approach to prevent this type of accident, by using one or more frontal cameras and sensors. The effectiveness of such a system depends on the time in which the possible accident has been detected prior to its occurrence. The main solution presented by Kovačić et al. [2013] is to use lane detection systems. Such systems are applicable in both ADAS, as well as in autonomous vehicles. By accentuating a dangerous lane change, the drivers attention can be focused using appropriate feedback.

Lane detection is often done in two parts. The first part includes processing techniques such as image enhancement and edge detection, whereas the second part extracts lane features and estimates the shape of the road. The main problems when implementing lane detection are incapability to detect lane markers, and the driver not being able to estimate the position of the vehicle in regards to the curvature of the road.

2.4.2 Lane Departure

From NHTSA [2008], it is evident that 22.2% of accidents in the US from 2005-2007 happened because vehicles were over the edge of the road, and in 10.8% vehicles overran the lane line. 3.1% happened after intentionally changing lanes or overtaking. Kovačić et al. [2013] argues that lane departure accidents can be significantly reduced by warning the driver through the use of computer vision systems.

The previously mentioned lane detection is one type of a driver assistance system, however, Kovačić et al. [2013] further argues that a driver fatigue detection is also a possible solution. Detecting driver fatigue could prevent thousands of accidents where the driver falls asleep, or is otherwise inattentive due to fatigue. Falling asleep while driving is often characterised by a gradual declining alertness, often due to monotonous driving conditions or other environmental factors. According to Kovačić et al. [2013], warning a fatigued driver one second prior to a possible accident situation, 90% of fatigue related accidents could be prevented. Driver fatigue detection can be implemented using various methods; analysing the drivers state (eyelids, head movement, gaze, etc.), driver performance (vehicle position and headway), or a combination of the two.

2.4.3 Crashes with Surrounding Vehicles

In regards to both frontal crashes, lane departures and in general, multiple vehicles are often involved in traffic accidents. Thus, tracking surrounding vehicles is relevant in many different applications. Comparing data from surrounding vehicles with the ego car state, or estimated future state, appropriate information and warnings can be presented to the driver to avoid accidents. Several sensors can be used for vehicle detection, such as motion sensors or depth cameras. Using optical cameras, however, can be quite challenging, due to both variability in vehicle appearance, as well as environmental factors such as weather conditions, which can reduce the effectivity of the image processing that is required. This is also related to the main problem of this type of system, which are the amount of false detection of surrounding vehicles, which could lead to wrong information being presented, and thereby endanger the driver. The approach towards vehicle detection is the same as in other object detection applications. First, candidates that could be vehicles are detected, before verifying the actual vehicles present.

2.4.4 Road Signilisation and Recognition

According to Kovačić et al. [2013], road signalisation represents the key element in traffic safety. Regardless of the context, the driver should be able to detect and recognise road signalisation, especially traffic signs. The signalisation, however, is often mostly ignored when the driver is unfocused or tired. Statistics from NHTSA [2008] shows that almost 110,000

accidents in the US were due to speeding and ignoring speed limits. For example, when driving in an area unknown to the driver, prioritising information importance and information overload is a big reason why certain signs are overlooked.

Recognition can be done due to key feature of an object. Two of the key elements of traffic sign appearance are their well distinguished colours and shapes, which are some of the aspects standardised and simplified by conventions such as the Vienna Convention on Road Signs and Signalisation standardised appearance of road signs throughout Europe and other countries [UNECE, 1968]. A more depth explanation can be found in Section 6.1.

2.5 Relaying Information Unobtrusively

After researching different solutions towards gathering traffic information, it is investigated how this information should be presented to the driver.

While driving, many senses are active simultaneously, and it is easy for the driver to be overloaded with information. Therefore, it is important to relay information to the driver in an unobtrusive way. The driver needs as much attention to the road as possible, thus the eye gaze should primarily be on the road. Jansen [ND] states that in order to create an unobtrusive interface, the primary modality used in the main task should not be the same used in the interface. In human-computer interactions, a modality is an input/output sensor that makes interactions between a computer and a human possible Karray et al. [2008]. This is because humans have multiple pools of attentional resources, and can divide their attention into several perceptual modalities. Humans are better at cross-modal attention sharing compared to inter-modal attention sharing. This means that if a driver's main task is looking at the road, then the interface should not rely on the visual channel. Matthews et al. [2004] states that the human attention can be categorised into three types; inattention, divided attention and focused attention.

A completely unobtrusive system is a system which requires no divided or focused attention. Based on the importance of the information, the more attention of the stimuli it should consume. Therefore, an interface in a car would most likely never be categorised as inattention, since most information has some level of importance, and at least require divided attention from the user.

To conclude, an interface which should not interrupt the important primary task of driving, should rely on other modalities than visual, if possible. There are situations that require the drivers immediate attention, and the visual feedback could be needed. A glance at a display could consume less than seconds, based on the position of it, and the visual channel is therefore still a viable choice of relaying information to the driver.

2.6 Field Research

Based on the research presented in Sections 2.2, and 2.3, a field research was conducted to investigate the aspect of driver information processing first-hand, to further support the theories presented. This section will present the different elements of the study, including its purpose and results; supported by visual examples.

2.6.1 Purpose and Setup

The purpose of this experiment was to get a first hand impression of:

- The broader context of the problem.
- What road information is considered most relevant by the driver.
- How aware the driver is of different road conditions and road information.
- Inattention, internal-, and external distractions.

Three pairs, a total of six participants were asked to navigate a pre-defined route through Aalborg Centre. The participants were 5 males and 1 female, between the ages of 21 and 25, and all students at Aalborg University. The 6km route consisted of different zones, surroundings, and traffic density. First, the route was driven by one participant while the other was navigating using a map, and afterwards the participants switched roles and the route was reversed. In both cases, the driver was unaware of the route. The route can be seen in Appendix A.1.

2.6.2 Results and Findings

The experiment was evaluated using observations, analysing video data, questionnaires, and qualitative semi-structured interviews with the participants. The major conclusions drawn from the experiment can be seen below, sorted according to the three factors mentioned by Shinar [2007b]; inattention, internal-, and external distractions.

Inattention

Several times throughout the test, various moments of driver inattention occurred. As mentioned in Section 2.3, familiarity with the environment can lead to driver inattention and lack of focus among. Several test subjects were used to driving in and around Aalborg, and drivers who were used to drive in the city carried out the task more calmly than those who were not. Furthermore, being familiar with the type of car also proved to affect the driver's stress-level.

Examples of inattention, were seen in different cases. Most drivers properly orientated while driving, however, a few situations occurred where they either orientated themselves to late, or were completely inattentive. One participant mentioned late instructions from the co-driver to be a factor in lacking/insufficient orientation. Sometimes, the co-driver supplied additional information, or guided the driver's attention towards what he might not have seen. This can be seen in Figure 2.7. Another participant missed a red light and continued through an intersection after the light changed. This was the most critical error seen in the experiment. The given situation can be seen in Figure 2.4.

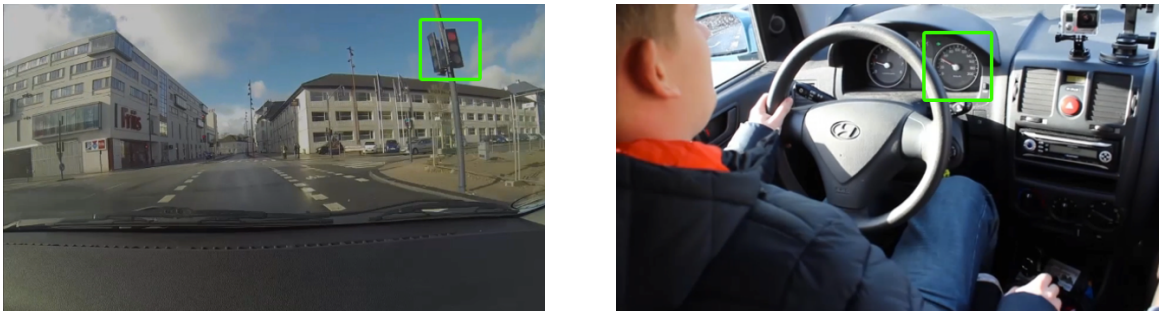


Figure 2.4: Driver misses red light and continues his right turn.

Internal Distractions

Throughout the test multiple test subjects were distracted by internal factors. Figure 2.5 shows the moment where the driver adjusts the rear mirror, just as she passes a speed sign. This resulted in her not properly adjusting her driving according to the sign information. Similarly, another driver was distracted by both side mirror adjustments and controls to defog the windows. In these cases the driver's attention was partially on the adjustments, however, it did not directly affect his driving.

Another internal factor was the communication between the participants. In some pairs it was seen that they were completely focused on the task, while others had consistent conversation and joking throughout the test. Common for all participants, however, was that all six drivers prefered clear concise instructions. Insecurity in instructions made the driver insecure about the directions as well. Only the following one or two upcoming directions was prefered, given well ahead of time. Instructions were given based on e.g. road names and objects in the environment. Gesturing and pointing were also used by most participants, used to supplement verbal instructions.

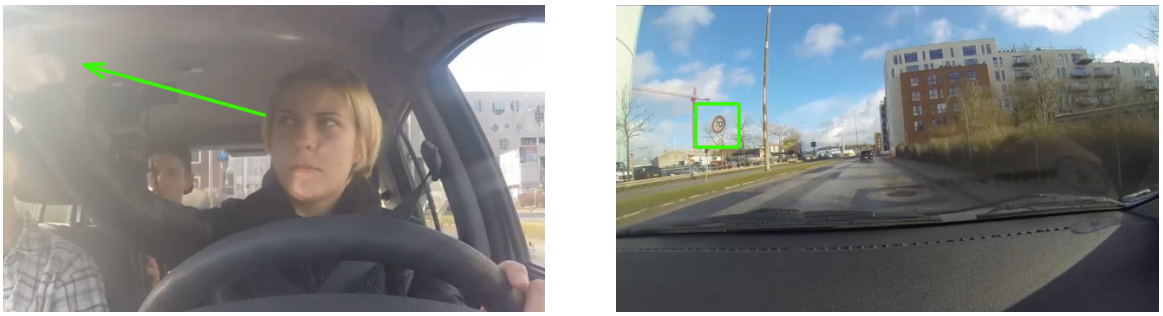


Figure 2.5: 70 kph sign is missed, as the driver adjusts the rear mirror while passing the sign.

External Distractions

On one particular location on the route, several test subjects missed a traffic sign which cancelled a 70 kph due to upcoming road maintenance, resulting in a new speed limit of 50 kph. As previously mentioned, being familiar with the environment can affect driver attention, which might have been the case here as well. Another possible reason was the positioning of the sign, which was immediately after a left hand turn. This situation can be seen in Figure 2.6.

Figure 2.6: Driver is speeding (40%) after missing the end of 70 kph-sign in what is a 50 kph zone.

The pairs who conversed most, talked a lot about the environment in which they were driving. In one instance the co-driver pointed out a billboard which directed the driver's attention away from the road. However, in some cases the co-driver actually guided the driver's attention towards relevant elements that he might have missed. An example hereof can be seen in Figure 2.7.

Figure 2.7: Co-driver guides the driver's attention towards the pedestrian crossing the street in the direction they are heading.

Another external factor was direct sunlight reducing the driver's vision in an intersection. This resulted in some insecurity about the surrounding traffic, and might have affected the driver's estimation and judgement hereof.

Other Findings

Three test subjects failed to correctly complete the given route, due to bad instructions and the co-driver misreading the map, current location, and entry allowances.

In regards to the interaction between the participants, all liked the possibility to get confirmation about correctly understood instructions. Most drivers mentioned having a co-driver or a GPS as their prefered method of being presented with instructions. This was due to both the visual and auditory feedback provided.

2.6.3 Conclusion

The findings from this field research did support the research previously presented in this chapter. It became evident that when driving, many different factors affects the driver in one way or another. This can result in either cognitive overload, less focus, stress, or inattention, as seen in various cases throughout the experiment.

Chapter 3

Problem Statement

As mentioned in Chapter 2.1, traffic accidents account for millions of deaths and injuries each year across the world. The DTSC have stated several areas which are of interest to help reduce the amount of accidents, one of which is technological advancement in terms of ADAS. Within the top four factors leading to traffic deaths, speeding is prevalent in 23% of cases and this number has remained constant over previous years. It can also be argued that the other three factors can lead to speeding, as more factors be present in accidents, these include lack of orientation, under the influence and lack of attentiveness. Further statistics from the US show that drivers that speed due to ignoring speed signs accounts for 110,000 accidents.

Driver habits were then researched, in order to receive a better understanding of the factors leading to so many accidents. Drivers have a lot of stimuli that is present and it is practically impossible to give 100% attention to it all. It is actually not necessary and in most driving cases it is completely fine, however, accidents occur when drivers are not attentive enough and cannot react quickly enough when unexpected events occur. There are many factors that can lead to a driver having a potentially slow and deadly reaction time. These include selective attention, which is relevant in terms of speed signalisation, where many drivers see speed signs, but choose not to react to them. Driver distractions and inattention also lead to a decreased reaction time, and it has been argued that if drivers were more attentive up to 80% of accidents could be eliminated.

To get a better understanding of these factors in general driving habit, six students from Aalborg were asked to drive a predefined route. Using video analysis, various factors were noted, such as inattentiveness, distractions and selective attention, confirming the research made prior to the experiment. Based upon the problems researched in the previous chapter, the following problem statement can be formed.

> *How can computer vision be implemented in an Advanced Driver Assistance System to recognise road signalisation, and how can this information be unobtrusively conveyed to the driver?*

Chapter 4

State of the Art

This chapter will introduce the concept of Advanced Driver Assistance Systems (ADAS). Some of the problems which may occur in the field of ADAS-development will be explained. Furthermore, state of the art in regards of ADAS and computer vision techniques will be described. The state of the art ADAS will be in regards of the three leading car manufacturers; BMW, Volvo and Mercedes-Benz.

4.1 Introduction to ADAS

Based on the research conducted in Chapter 2, it is concluded that many traffic accidents can be associated with human error. There have been many initiatives throughout the years which have researched the problem thoroughly, researching solutions to minimise traffic accidents. One such initiative is Prometheus, which was a series of projects conducted by European car manufactures and research institutes, which started in 1986. Shortly after, the European Union initiated the DRIVE (Dedicated Road Infrastructure for Vehicle safety in Europe). These initiatives led to the start of ADAS. The main purpose of ADAS was to eliminate or reduce the amount of human errors while driving a vehicle, whereas the future goal would be to make the cars self-driving, completely eliminating the impact of human decisions.[Brookhuis et al., 2001]

4.1.1 Problems with ADAS

There are, however, some problems when adding ADAS to vehicles. As mentioned in Section 2.3 one of the leading factors to traffic accidents is inattention and distraction. The attentive level required while driving decreases as the automation of driving increases, which might lead to a lower general driver alertness. If the design of the system is not properly intuitive, it would increase the complexity of driving, which increases the likelihood of failure by the driver. [Brookhuis et al., 1993]

In a test evaluating different types of collision avoidance systems, drivers said they expected an increase in safety when using systems which took control of their driving. One such situation is when emergency brake is necessary, but they would still have a hard time giving up the control of the vehicle. [Nilsson and Alm, 1991]

A global survey was conducted during project SAVE, which is a project aimed for creating ADAS that takes control over the vehicle in emergency situations. The survey concluded that drivers were reluctant to release control of the vehicle, but would be willing to accept it in situations of emergencies, such as sudden illness while driving. [Brookhuis et al., 2001]

The following section will list the state of the art in ADAS.

4.2 Interfaces and Modalities

In order to convey information to the driver, the interface must be created in an unobtrusive way. This section will describe state of the art feedback systems implemented in cars. According to ABIResearch [2015], the top three manufacturers of ADAS are:

- BMW
- Volvo
- Mercedes-Benz

Those three accounts for at least 50% of consumer vehicles sold in Western Europe in 2014 with at least one form of ADAS [ABIResearch, 2015]. This section will look into those three manufacturers. Some of the different ADAS will be described, with a focus on how the information is perceived by the driver.

4.2.1 BMW

BMW has developed a computer system for their cars called iDrive. The system consists of a display placed in the middle top of the dashboard, and a controller placed next to the gear knob. The iDrive controls most settings inside the car, such as navigation, radio, door locks, phoning, car info and more. When released it had haptic feedback in the controller, such that the driver could pay more attention to the road while interacting with the system. This feature was removed in 2007, due to responses from customers saying it was annoying. BMW uses ADAS, such as forward collision, city brake, pedestrian detection, parking assistant, blind spot detection. The way information is conveyed to the user is through 3 modalities: visual, auditory and haptic. Visual in form of the previous mentioned display, but in newer models a heads-up-display (HUD) is implemented. An LED in the side mirror flashes when there are cars/obstacles in the blind spot. An auditory feedback based on the importance of the alert, and a haptic feedback in form of a vibration in the steering wheel. Examples of ADAS systems in BMW cars can be seen in Figure 4.1. [BMW, 2015a][BMW, 2015b]

(a) HUD concept. [BMW, 2015a]

(b) iDrive display. [BMW, 2015b]

Figure 4.1: State of the art ADAS in BMW cars.

4.2.2 Volvo

Volvo has created a system called Intellisafe, which uses multiple ADAS, such as cyclist and pedestrian detection, parking assist, blind spot information, forward collision warning, and lane departure. These systems also utilise various modalities. One example is forward collision warning, which checks if there are any obstacles 500 feet in front of the vehicle. If the obstacle is in danger of the driver, the system both delivers a visual feedback in form of a display in the dashboard as well as a HUD, and an auditory feedback is given in form of a beeping sound. Lastly a haptic feedback is applied to ensure the attention of the driver. If the driver does not react on the warning signals in time, the Intellisafe takes control of the vehicle, and performs a full brake, to ensure collision avoidance. Example of the ADAS HUD in a Volvo can be seen in Figure 4.2. [Volvo, 2015]

Figure 4.2: HUD in a Volvo car. [Volvo, 2015]

4.2.3 Mercedes-Benz

Mercedes-Benz also uses multiple ADAS, much like Volvo and BMW. They call the system Mercedes-Benz Intelligent Drive. The two main modalities utilised are auditory and visual feedback. Unlike volvo and BMW, Mercedes-Benz have placed their main visual feedback on the dashboard instead of a display. An example of an ADAS is Attention Assist, which can determine if the driver is getting drowsy and unattentive, based on up to 70 measured

variables. If that is the case, a visual feedback will appear in the dashboard, while an auditory feedback alerts the driver, informing him or her to take a break (see Figure 4.3a).

Another example is their Active Blind Spot Assist, which will check for objects in the blind spot using sensors. If an object is detected, a visualisation will pop up in the dashboard, as shown in Figure 4.3b, and auditory feedback will be played in form of a beeping sound. If the driver activates his/her indicator light toward that direction, a more intense sound will be played. And if the car is in danger of collision, it can gently guide the car back to the lane. Some of the Mercedes cars uses haptic feedback in the gas pedal, to give information to the driver. One example is if the driver goes too fast, small vibration patterns will be delivered to the foot. Or when a hybrid car switches from gas to electricity, a different pattern is sent to the driver. [Mercedes, 2015]

(a) Attention Assist visual feedback.

(b) Active Blind Spot Assist visual feedback.

Figure 4.3: The ADAS implemented by Mercedes. [Mercedes, 2015]

4.2.4 Conclusions

The leading manufactures of ADAS in Western Europe use similar technologies. Most of the interfaces are alike, and the feedback is given using the same modalities; visual, auditory and haptic. The visual feedback differs in animation and placement, for example in form of light indicators placed at the side mirrors. The auditory feedback is used in the same way as alerts, but with different sounds. Haptic feedback is used differently between the manufactorers, normally through the steering wheel for extra feedback, but is also used as minor feedback when interacting with car controls.

These three types of feedback were investigated by Meng and Spence [2015] and Stanney et al. [2004]. Based on these investigations, advantages and disadvantages of the modalities when used inside a vehicle, have been collected in Table 4.1.

	Advantages	**Disadvantages**
Visual	- Straightforward means of conveying information - Fast reaction time	- Competing with the visual resources for vehicle control. - Can easily be missed while driving.
Auditory	- Easy to convey spatial information by verbal signals. - Perception of auditory stimuli is gaze-free. - Fast reaction time.	- Easily masked by background noise or be interfered with by secondary tasks. - Can be difficult to localise the spatial direction from which the auditory warning signal is presented. - Some drivers suffer from hearing impairment.
Haptic	- Less central to driving. - Does not increase visual or auditory workload. - Less interfered with by secondary tasks while driving. - Direct interface to the driver. - Convey alerts and warnings.	- Limited amount of interfaces to deliver the feedback. - One-handed driving, clothing/ gloves, and in-vehicle vibrations might impair effectiveness.

Table 4.1: Advantages and disadvantages of visual-, auditory-, and haptic feedback. [Stanney et al., 2004][Meng and Spence, 2015]

From Table 4.1, it can be seen that each modality has strengths and weaknesses, and vary in ability to convey different types of information. This will be explained more in Section 9.2.

4.3 Traffic Sign Detection Methods

This section will describe some of the possible solutions in detection of traffic signs. The solutions can be divided into three categories:

- Colour-based detection
- Shape-based detection
- Learning-based detection

Each category will be described below which is based on the researched conducted by [Brkić, ND].

4.3.1 Colour-Based Detection

Traffic signs have been created in a way that they are easy to spot with the human eye. This means that the colors have been chosen such that there should be a high contrast to the background. Colour-based detection then uses this colour information, in order to threshold

the image with that specific colour, in order to detect the sign. There are, however, some complications regarding this. The weather and illumination can have such a great impact on the colour, that the bright red might now be seen as a very dark red, due to shadows. The standard RGB colour space will have a hard time distinguishing the difference. Colour spaces that use an illumination component can then be used, such as HSI, L*A*B, CMYX, and LUV. These colour spaces can improve the detection rate, however the detection rate could still be diminished under the impact of bad weather. Another pitfall regarding colour-based detection is the fact that there could be other objects in the image, with the same or similar colour as the traffic sign, and would therefore produce a false positive. Colour-based detection can however still be used as a method of determining the region of interest, in which other methods can be applied, in order to prevent those false positives.

4.3.2 Shape-Based Detection

Another feature is the shape of the signs. There are several ways of detecting shapes in an image, among the more popular choices are template matching, and corner detection. As well as the colour-based detection, shape-based detection can deliver a high detection rate, but with the cost of false positives. If the detector is looking for square signs, then it might as well also find a truck that is driving in front of the car, since it has a square shape as well. And due to the fact that traffic signs are of fairly simple shapes, the chances of other objects with those simple shapes are relatively high. This means that the detector should be used in collaboration with other methods, just as the colour-based detection.

4.3.3 Learning-Based Detection

The learning-based detection, also known as machine learning, uses the two methods described above. The principle of the method, is that a computer is given a set of training data, which consist of many images. Each image is then annotated if there is a sign on it, or if there is not. The computer then runs through each image, and learns patterns of features in the images. When the training is done, the computer should be able to determine how a traffic sign looks, and can then detect new signs afterwards. The method was first invented for face-recognition, but is widely used for detecting all sorts of objects. Viola and Jones [2012] presented their work with machine learning back in 2001, and that has later been a milestone within computer vision. The method has been improved since then, but the basic theory remains the same.

Conclusion

The computer should be trained in detecting signs, but in order to improve the flow of the system, either colour-based detection or shape-based detection, or a combination, should be used to find region-of-interests, where there might be a traffic sign, and then learning-based detection should be applied to those specific regions, in order to reject the false positives. Section 6.2 provides a more in-depth description of learning based detection.

Chapter 5

Requirement Specification

Based upon the problem statement in Chapter 3 and state of the art research in Chapter 4, this project will focus on the design and implementation of an ADAS capible of detecting speed signs, and relaying relevant warnings to the user. The purpose of creating a requirement specification is to state a set of use cases where the system will function. A use case is a representation of user's interaction with the system, and in order to take the users needs and requirements into consideration, the functional requirements will be stated. These requirements needs to be accounted for in the whole system, but due to time constraints of this project, only a few of these requirements will be implemented and tested. In order to evaluate the design and implementation of such an ADAS, a specific set of goals need to be stated.

5.1 Actors

In a use case, an actor models a type of role played by an entity, that interacts with the system in some way. It is not necessarily human, but can also be an external element which exchanges signals and data. This project has the following actors:

- **Driver:** The driver of the car.
- **Camera:** A camera which is able to detect speed limit signs.
- **Speed limit sign:** The speed limit sign which needs to be detected.

5.2 Use Case

The system has a limited amount of possible human interactions, therefore only a single use case will be described. This section contains that use case of the system, with a purpose to identify the functional requirements. The use case diagram can be seen in Figure 5.1.

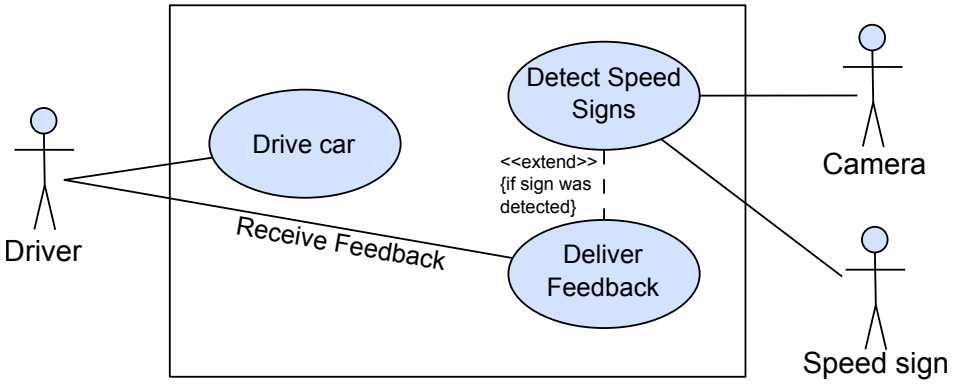


Figure 5.1: Use case diagram for the system.

The three actors that can interact with the system are; the driver who controls the car, the camera which can detect the third actor; the speed sign. Based on this the system gives feedback, if the car is exceeding the speed limit, to the driver.

- **Actors:** Driver, camera, speed limit sign.
- **Start Conditions:** Initiated when the camera detects a speed sign or when the driver is speeding.
- **Successful Result:** The driver slows the car to the speed limit.

Two scenarios in which this use case occurs is outlined below.

Normal Scenario

1. The camera detects a speed sign.
2. If the driver is driving above the speed limit, feedback is given to the driver.
3. Driver decreases speed of the car to the speed limit.

Alternative Scenario

1. The system has previously detected a speed sign and stored this information.
2. If the driver is driving above the speed limit, feedback is given to the driver.
3. Driver decreases speed of the car to the speed limit.

The use case outlined above is a simple scenario and assumes that the user reacts to the feedback that is given. As the system does not force the user to decrease the speed limit of the car, an analysis of different feedback methods will be done in Section 9.2.

5.3 Functional Requirements

The use case is an example of how the various actors interact with the system, but does not show what the requirements are for the system to fulfil those actions. Therefore the following section will outline those requirements for a functional system.

1. A camera must be able to capture road information.
    - The camera must have a suitable size for a car.
    - The resolution should be 640x480, due to the fact that the potential traffic sign could be detected in a relevant distance, and the processing time would be sufficient.
    - Traffic sign should be detected at a size of minimum size of 20x20, in order to be detected from a great distance. The size may not be too low, due to lack of detail.
    - The camera must have a minimum frame rate at 5 frames per second (fps), optimally 25 fps. The minimum frame rate is based upon the need of detection. The traffic sign is just required to be detected once before passing, therefore the frame rate is not required to be relatively high. The camera should however not limit the flow of the system, hence the optimal frame rate.
2. The system must be able to detect speed signs.
    - The detector needs to run with a minimum of 5 frames per second.
    - The current speed limit must be stored in the system until a new speed limit has been detected.
    - The system must detect speed signs as a minimum in 1 out 5 frames.
    - The system must be able to classify the speed signs, in order to determine the speed limit on the given road.
3. The system must be able to store information regarding the velocity of the car.
    - Access to the car velocity must be gain from the cars integrated velocity sensor.
4. The system must be able to give three different feedback types to the user.
    - Haptic feedback; vibration in the steering wheel.
    - Visual feedback; a heads-up display.
    - Audio feedback; sounds though the integrated car speakers.
5. The speed signs follow the Vienna convention, which is explained in Section 6.1, and have the following attributes:
    - The shape of the sign is circular.
    - The colour of the sign is white, with a solid red border which covers 33% of the entire sign.
    - The centre of the sign must state the speed limit with easily readable numbers.

5.4 Context Requirements

The context requirements is an addition to the functional requirements. The functional requirements only state the requirements in a normal situation. However, special cases might arise due to the time of day, weather conditions and other special cases. The developer must be aware of the following contexts, in order for the system being able to handle them:

Weather condition: The weather needs to be taken into account, due to the fact that it changes throughout the year. Special weather conditions have a great impact on the ability to read road signalisation. Such weather conditions could be snow, rain, fog and bright sunshine.

Time-of-day: The lighting changes throughout the day, and this needs to be taken into account, especially near dusk and dawn.

Perspective of sign: The angle towards the road signalisation will change while approaching. This will change the shape of the sign slightly, based on the perspective from the camera.

These are the context requirements which will be taken into consideration in this project. Based on these the system should be able to detect speed limit signs with response to the functional requirements, and the context requirements should be taken into account when implementing it. Ideally the system should meet these expected requirements.

5.5 Project Outline

The functional requirements defined above outlines how the complete system should be implemented. The aim of this project will not be to directly implement this system, but rather investigate the two main aspects the systems have; the detection of speed signs, and interface with the driver. Furthermore, general observations of driving behavior is of interest. To investigate and evaluate on these, the requirements for the system implemented in this project will be to:

1. Detect, but not classify, speed signs in a single image.
2. Run a detection benchmark, to evaluate on the detector.
    - Have a detection rate of 20%.
3. Present the driver with visual, auditory, and haptic feedback.
4. Evaluate the interface regarding:
    - Intuitive understanding of feedback.
    - Reaction time.
5. Observe general driving behavior with respect to:
    - Selective attention.
    - Distractions.
    - Inattention.

The following Chapter will present a computer vision, machine learning algorithm which can be used to detect speed signs, whereafter implementation and evaluation hereof will be presented. The detector evaluation will be investigated in Section 8.3. Chapter 9 will present the design and evaluation of the interface.

Chapter 6

A Method for Traffic Sign Detection

This chapter will present information about the objects that need to be detected; traffic signs. Furthermore, relevant theories in regards to implementation of a detection algorithm, Integral Channel Features (ICF), will be explained in-depth. Different aspects of ICF will be described; channels, features, boosting, and classification.

6.1 Traffic Signs

Before going more in-depth about the detection of traffic signs, the design of the signs themselves needs to be assessed, which will be presented in this section.

With the reduction of traffic accidents in mind, signs play an important role in traffic safety. They are designed to be salient, and convey road information and traffic regulations to drivers, in a simple and comprehensible way. To do so, several aspects must be considered. First of all, the signs must be visible to the driver. According to Porathe and Strand [2011], visibility concerns different object properties, such as size, colour, contrast, and brightness. They also state that it is important to keep the surroundings and external conditions in mind when designing traffic signs. These two factors can limit the saliency of the signs, due to factors such as the varying weather- and lighting conditions. Furthermore the signs need to be visible in different driving environments, both in cities with dense traffic, and on open country roads. Wang et al. [2013][Porathe and Strand, 2011]

6.1.1 Standardisation of Traffic Signs

Across the world, traffic signs differ of varying degrees which impacts the comprehensibility when driving in countries where certain information is conveyed differently. To avoid such confusion, various standardisations have been made, such as the Vienna Convention on Road Signs and Signals, which standardised traffic signs and signalisation internationally, especially among European countries [UNECE, 1968]. As per 2014, a total of 63 different countries have incorporated these regulations or an amendment hereof [United Nations, 2015].

The convention categorises different types of signs by shape, where circular signs are prohibitions including speed limits, triangular signs illustrate warnings, and rectangular signs defines recommendations or subsigns. Furthermore, octagonal signs signal full stops, and downward-pointing triangles signals a yield. Examples of road signs as per the Vienna convention can be seen in Figure 6.1. [Møgelmose et al., 2012][UNECE, 1968]

(a) 60 kph speed limit. (b) Right turn. (c) Stop sign.

Figure 6.1: Examples of Danish road signs as standardised by the Vienna Convention on Road Signs and Signals.

6.1.2 Challenges

As mentioned, variations in conditions and complexity of outdoor environments challenges the recognition of traffic signs, even though their appearance have been standardised. Wang et al. [2013] and Møgelmose et al. [2012] list some of the main challenges faced by traffic signs recognition systems, some of which have been covered in Section 5.4:

- Variations in lighting conditions depending on time, season, and weather.
- Reduced colour contrast, due to similarity between sign colour and background colour.
- Clustered traffic signs.
- Occluded or partially occluded signs.
- Viewing angle may result in signs appearing rotated or deformed.
- Sign condition; could be dirty or faded.
- Similar objects might appear in cluttered environments.

Since this project aims to detect speed signs, standardisation of signs makes the task somewhat easier, as a learning-based detection method requires a large data-set of images. As this may not be available for a specific country, this standardisation means that it can be possible to find datasets from other countries.

6.2 Integral Channel Features Algorithm

Different methods for detecting objects in a two-dimensional image exist, one of which is Integral Channel Features (ICF). Dollár et al. [2009] presented the ICF method for object detection. The method was designed for pedestrian detection, but as research shows, however, this method can also be used to detect other objects.

ICF is one of the current state of the art pedestrian detectors. The algorithm has been tested on the Inria and Caltech data-set, which is a set of images with pedestrians. To test detectors, data-sets containing images of pedestrians has been created, and running the detector on these images gives a benchmark for the detector. The detectors is evaluated on the amount of correct detections, called true positives, and wrong detections, called false positives. ICF performs second best in the Inria data-set, with a detection rate of 86% with 1 false positive per window, and with the Caltech data-set is has a detection rate of 60% with 1 false positive per window. The results of the tests combined with the simple implementation makes ICF the detector which will be implemented in this project [Dollár et al., 2009]. The ICF algorithm has later made ground for developing the Aggregate Channel Features (ACF), the ACF algorithm improves the computation time compared to ICF, making it faster to run detections [Dollár et al., 2014]. Since ICF was the building block towards ACF, it was decided to implement an ICF in this project.

The ICF algorithm consists of five blocks, these blocks are illustrated in Figure 6.2. The first block is to *Calculate Image Channels*. Channels are different representations of the same input image. Common channels are the colour channels R, G and B which combined gives a colour image. Figure 6.3 illustrates these three colour channels. Section 6.2.1 discusses the possible image channels, and later in this chapter the calculations of the chosen image channels will be explained.

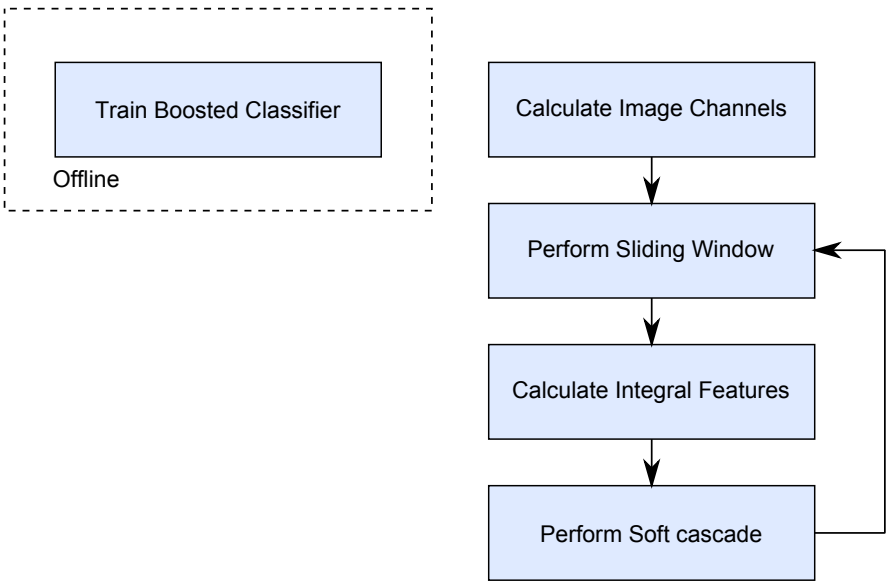


Figure 6.2: The main program algorithm.

(a) Combined RGB image. (b) Red channel. (c) Green channel. (d) Blue channel.

Figure 6.3: A illustration of different colour channels, (b), (c) and (d) are each a image channel, combing these channels results in a RGB images which is illustrated in (a).

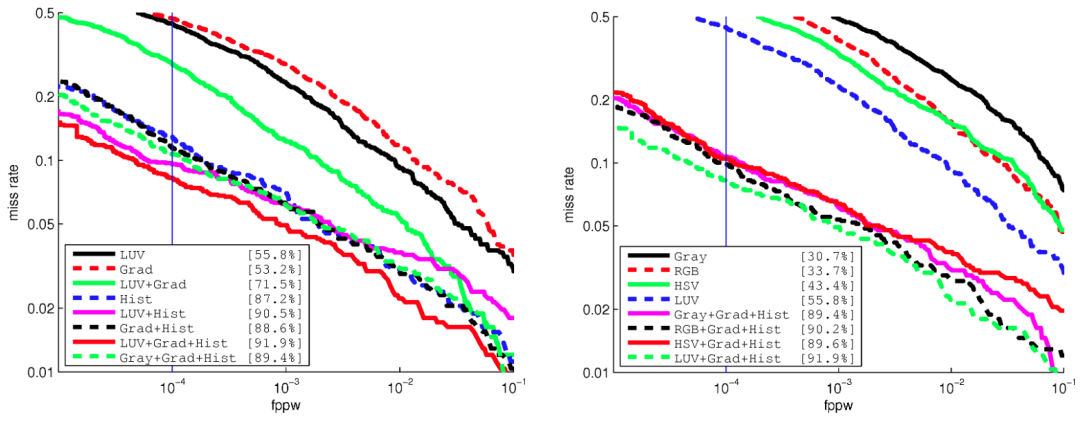
The second block in the algorithm is to *Train Boosed Classifier*. A boosting algorithm can generate a set of weak rules, that can choose the features that are best at separating the objects from everything else. Doing this requires a training data-set. The training set consists of images only showing the object of interest, called positive images. A negative data-set, which does not include images of this type of object, is also required. This part of the algorithm is done separately from the rest of the program, thus labelled *offline* in Figure 6.2. This is explained further in Section 6.4.

The third block in the algorithm is *Performing Sliding Window*, to find the position of a road sign in a large image a window is slided through the image. The window creates the scope for the classifier. Inside each window the following blocks of the algorithm is performed: *Calculate Integral Features* and *Perform Soft cascade*. *Calculate Integral Features* is done on each of the image channels. In the algorithm suggested by Dollár et al. [2009], first order features are introduced. First order features are the pixel sums of local squares in each image channel. To improve on computation time, integral images are used for calculating the local pixel sums. An integral image is a representation of an image where pixel (x, y) is the sum of all pixels above and to the left [Viola and Jones, 2001]. Calculations of the local pixel sums will be explained in Section 7.1.3. The last block in the algorithm is performing a soft cascade. The soft cascade is generated by the boosting algorithm, and decides if the object, which it has been trained to detect, is inside the window. Having performed the soft cascade, the sliding window continues. The soft cascade classifier is explained in Section 6.5 and the sliding window is explained in Section 7.2.2. These different elements of ICF will be explained in the following section.

6.2.1 Calculate Image Channels

For an ICF detector, image channels are required for calculating features. A large variety of channels exist, such as grayscale image, RGB channels, and more complex channels [Dollár et al., 2009]. This section investigates which images channels performs best, based on research conducted in this area.

Tests are conducted by Dollár et al. [2009] on both the INRIA pedestrian dataset and the Caltech pedestrian dataset. The evaluation shows that channels in conjunctions outperform single channel features. The colour spaces gray, RGB, HSV and LUV were evaluated. The study showed that LUV outperform the other colour spaces. The evaluation of the detection rate is done taken at a false positives per window(FPPW) rate at 10^{-4}. This describes the amount of false detections per window. The LUV colour space has a detection rate of 55.8%, and the second best colour space, HSV, has a detection rate of 43.4%. Combining channels, however, can improve the detection rate significantly. Dollár et al. [2009] made several combinations of channels to find the best combination. Combining the gradient magnitude channel with the gradient histogram channels results in a detection rate of 88.6%. This is the best performance for a system combining two channel types. This can be improved on by adding a third channel type but the performance gain is far smaller going from two channels to three than going from the single channel to two channel types. The best performance from combining three channel types is reached by combining the LUV colour space, gradient magnitude channel, and gradient histogram channel. The combinations of these three channel types has a detection rate of 91.9% which is the highest detection rate. In Figure 6.4 ROC are shown for different combinations of channels, and their detection rates across FPPW.



(a) (b)

Figure 6.4: ROC tables from Dollár et al. [2009], evaluating ICF channels. (a) Showing combinations of channels and (b) adding 4 colour spaces for comparison.

These results give support for choosing the channels for creating a detector for this project. For the best performance possible the channels that will be used for this project will be the best combinations of channels from the paper, which is the combination of LUV, gradient magnitude and gradient orientation, which will be explained in the following sections.

LUV Colour Space

In order to perform a conversion from RGB to LUV, the RGB must first be converted to the XYZ colour space. The XYZ colour space originated in 1931 when the International Commision on Illumination (CIE) defined three primaries, XYZ, which could be used instead of the traditional RGB-components. The reasoning behind this, was that not all visible colours could be shown using RGB-values, however, this was now possible with the newly formed XYZ-space. The colour space was defined so that all colours could be visualised using only positive values. The Y-value is the luminance of the colour [Oregon Institute of Technology, ND]. Converting an RGB image to the XYZ colour space is done by using a simple matrix calculation

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{6.1}$$

which results in the new colour space [OpenCV, 2015a].

CIE-LUV was formed in the attempt to create a perceptually uniform colour space. Therefore the distance between points could indicate the difference in luminance, chroma and hue between colours [Schwiegerling, 2004]. The calculation from XYZ to LUV colour space is computed as follows. Let X_n, Y_n, and Z_n be the reference white point in the colour space of a standard D65, where $X_n = 0.950456$, $Y_n = 1$, $Z_n = 1.088754$. Define,

$$u' = \frac{4X}{X + 15Y + 3Z} \qquad v' = \frac{9Y}{X + 15Y + 3Z} \tag{6.2}$$

$$L = \begin{cases} 116(\frac{Y}{Y_n})^{\frac{1}{3}} - 16, & \text{if } \frac{Y}{Y_n} > (\frac{6}{29})^3 \\ (\frac{29}{3})^3 \frac{Y}{Y_n}, & \text{if } \frac{Y}{Y_n} \leq (\frac{6}{29})^3 \end{cases} \tag{6.3}$$

then calculate

$$U = 13L(u' - u'_n) \qquad V = 13L(v' - v'_n) \tag{6.4}$$

where the values of u'_n and v'_n are the chromaticity of a specified white object, which in this case is $u'_n = 0.198$ and $v'_n = 0.468$ [OpenCV, 2015a]. Now that the RGB components of the image have been converted to LUV space, three channels have been computed in which ICF can be calculated. The individual components of an image can be seen in Figure 6.5.

(a) L channel. (b) U channel. (c) V channel.

Figure 6.5: Example of the LUV channels.

Gradient Magnitude

Another commonly used channel is gradient magnitude. Edges in images are defined as positions where significant changes in gray-levels occour [Moeslund, 2012]. In image processing, these edges are found using gradients in the image. Whereas the gradient of a point in a one-dimensional curve corresponds to the slope of the tangent at this point, the gradient in an image has two gradients, in the x- and y- directions. The gradient in the x-directions is referred to as G_x and G_y in the y-direction, these can be derived from a image by the use of kernels. Such two kernels are illustrated in Figure 6.6.

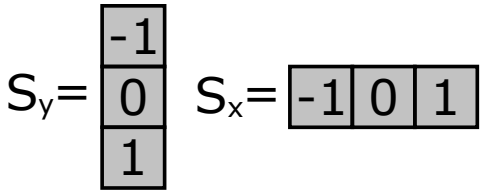


Figure 6.6: Vertical kernel G_x and Horizontal kernel G_y.

Correlating the kernels with the input pixels results in G_x and G_y which together can form a gradient vector

$$\vec{G}(g_x, g_y) \tag{6.5}$$

$\vec{G}$ then lies on the tangent plane at the given point, (x, y), and can be considered as the direction in which the changes in intensity are the highest. This can be visualised in a 3D space, using the x,y-dimensions of the image and by interpreting intensity values as height values. Figure 6.7 shows this visualisation, as well as the three gradients in (x, y). [Moeslund, 2012]

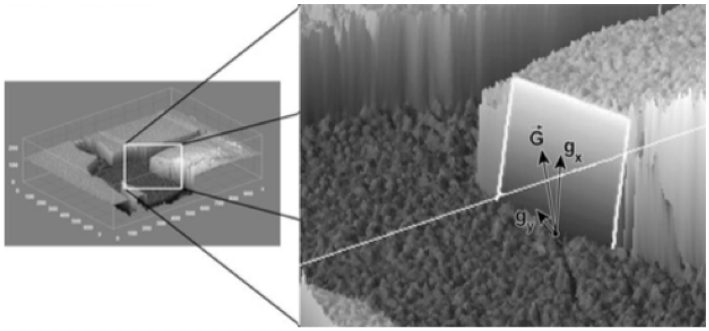


Figure 6.7: A 3D representation of an image, with gradients (g_x, g_y) as well as the gradient vector $\vec{G}$ in point (x, y). [Moeslund, 2012]

The length of the gradient vector can be determined by calculating the magnitude of $\vec{G}$, using

$$|\vec{G}| = \sqrt{(g_x^2, g_y^2)} \tag{6.6}$$

where $|\vec{G}|$ is the length of the vector. To decrease computational power, the magnitude can be approximated instead, with

$$|\vec{G}| = |g_x| + |g_y| \tag{6.7}$$

6.2.2 Gradient Orientation

According to Dollár et al. [2009], gradient orientation is the best performing ICF channel. This channel is based on Histogram of Oriented Gradients (HOG). The basic principle of HOG is that objects have a shape, and within this shape, there is a lot of relevant information. This includes the orientation of the edges of an object. The edge of a circle have an orientation or direction distributed from 0-360 degrees, whereas a square only has four different orientations. Moreover, each object have their own set of edge orientations. Dollár et al. [2009] uses this method to create individual image channels consisting of gradient directions. In the HOG algorithm, angle bins are created, consisting of given ranges of angles between -π to π or 0 to 2π. At this point, Dollár's implementation differs from the HOG algorithm. The gradient histogram channel weights the the histogram bins with the gradient magnitude, and maps the pixels with a gradient angle inside the given ranges to an output matrix. In principle, infinitely many channels could be created from a single image using this algorithm. Dollár et al. [2009] performed tests to evaluate how many angle bins are most efficient, which showed that six angle bins perform equally with 12 bins, which makes the choice about system performance. Six angle bins requires half as many computation thus makes the best choice. Figure 6.8 illustrates the channels generated with gradient orientation and the steps before the gradient orientation channels. The gradient orientation channel uses the gradient magnitude to weight the the orientations. The algorithm is as follows

$$Q\theta(x, y) = G(x, y) \cdot \mathbf{1}[\Theta(x, y) = \theta] \tag{6.8}$$

where $Q\theta(x, y)$ is the weighted angle, $G(x, y)$is the gradient magnitude, which is the factor the angle is weighted with. $\Theta(x, y)$ is the angle of the pixel, the calculations for the angle will be illustrated below. The indicator function is 1 if the calculated angle is equal to θ, which is the angle bin index. If this statement is false the indicator function is 0. This determines if the pixel in input image $I(x, y)$ gets mapped in the output, as the gradient magnitude.

Calculation of $\Theta(x, y)$ can be done using the vertical and horizontal edge kernels. The kernels are illustrated in Figure 6.6 in Section 6.2.1. The orientation of the gradient at $\Theta(x, y)$ can be computed using

$$\theta = tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{6.9}$$

where G_y is the output at $I_{x,y}$ after correlating with the horizontal kernel and G_x is the output after correlating with the vertical kernel.

After having calculated all channels, the features can be calculated randomly on the channels which will be explained in the following section.

(a) Input image. (b) Gradient Magnitude. (c) Orientations from 0-60. (d) Orientations from 60-120.

(e) Orientations from 121-180. (f) Orientations from 181-240. (g) Orientations from 241-300. (h) Orientations from 301-360.

Figure 6.8: Illustration of seven channels including the input RGB image. (a) shows the input image, which is processed to seven different channels. (b) shows the magnitude gradient channels derived from the input image. (c-f) shows six orientation channels.

6.3 Calculating Integral Features

The step in the algorithm after calculating image channels is to calculate the features on those channels. In order for a system to be able to detect an object, a set of features needs to be calculated. Integral features will be calculate on individual windows in an image, this set of features is what makes the window unique compared to other windows. The features makes it possible for a classifier to detect if a specific object is inside the window. Integral features are calculated from the channels, by taking the local sum of rectangles. The rectangles are randomly generated and should have a defined minimum size. Taking the local sum of a rectangle generates a first order feature. Higher order features can be generated by combining the local sum of two or more rectangles. [Dollár et al., 2009]

The ICF method uses the algorithm described above, but improves upon the calculations for the local sum. Using integral images, the rectangles described above can be computed rapidly, using an array of four references to calculate any given size rectangle, this is illustrated in Figure 6.9. [Viola and Jones, 2001]

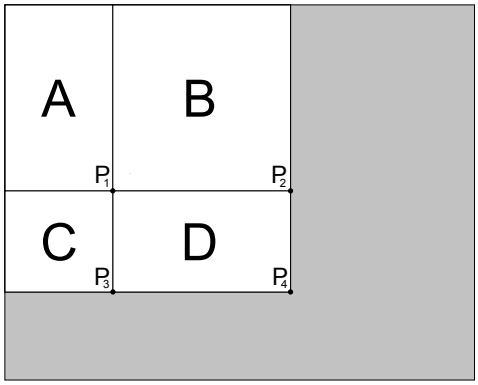


Figure 6.9: Sum of rectangle D can be computed with $p_4 + p_1 - (p_2 + p_3)$. [Viola and Jones, 2001]

Using the method described by Viola and Jones [2001], the pixel sum of rectangle D in Figure 6.9 can be calculated using an integral image. In this case, D is a representation of the integral image. D can be computed based on its four corners, denoted p_1, p_2, p_3, and p_4. p_1 is the pixel sum of rectangle A, p_2 is the pixel sum of rectangles $A + B$, p_3 is the pixel sum of rectangles $A + C$, and p_4 is the pixel sum of rectangles $A + B + C + D$. To take into account that rectangle A is part of both p_2 and p_3, p_1 is added to p_4 before subtracting the two. D can then be calculated from the pixel sums using

$$p_4 + p_1 - (p_2 + p_3). \tag{6.10}$$

Features calculated on each channel using the integral image are called ICF. ICF is very efficient because of the sum calculations done on integral images, and is therefore the prefered method for faster detection [Dollár et al., 2009].

6.4 Boosting

This section will describe the theory behind boosting, and why it is relevant when implementing for this algorithm in terms of object detection. As stated in the main algorithm in Section 6.2, the boosting algorithm performed offline from the rest of the program. However boosting requires features from positive and negative images, and it is therefore necessary to use the blocks *Calculate Image Channels*, *Perform Sliding Window*, and *Calculate Integral Features*.

Once the larger number of random features have been computed using ICF over both a positive and negative set of images, a technique called boosting can be used to create an accurate prediction rule. Boosting is a method which aims to improve the accuracy of different learning algorithms, where a number of innacurate rules, in this case the features computed by ICF, can be combined to an ensemble classifier Freund and Schapire [1999]. These innaccurate rules, can be defined as weak learners and need only to be better than chance, above 50%, where a boosting algorithm can pick the best weak learners and place weights upon them, depending to their importance to the algorithm. Kun [2015], states that

boosting is an example of a Multiplicative Weights Update Algorithm (MWUA), which has many applications, including learning algorithms. The idea of MWUA is to:

1. Maintain non-negative weights for elements of a set of data.
2. Take a random element of the data proportional to the weights.
3. Apply an algorithm on the chosen element based on outcome of another factor.
4. Update the weights and repeat the process.

Point three is where most of the interesting parts of boosting occur. The output of this algorithm is either a reward or a penalty on the randomly chosen element. This is done by adjusting the weights for the elements according to how the element differs to the goal, for example how much a given feature differs from the specfic feature of a speed sign from an image channel. The final objective of boosting is to minimise the regret on the dataset with respect to the best alternative.
Boosting uses a large number of weak learners to produce a strong learner. The final goal, the strong learner, is an overall rule that can be applied to data, to classify it as a speed sign or not. This is computed with a boosting algorithm with weak learners, which is essentially a rule of thumb and needs only to give correct results at a rate of more than 50%. Boosting can use the weak learners as binary classification, one method of doing this is to with decision stumps. Figure 6.10, shows an example of a decision stump on a two-dimensional feature space, where the features to the above in the red area, are given an output label -1 as they are considered negatives, where features below are given +1 as they are considered positives. In the figure, blue crosses are positive instances, whereas red squares are negatives. [Kun, 2015]

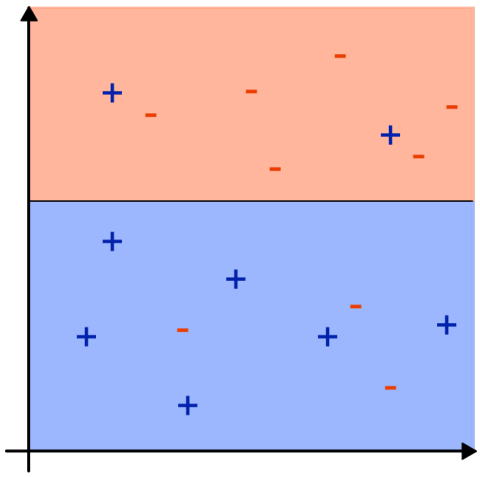


Figure 6.10: Example of a feature space with a decision stump placed horizontally.

The decision stump is then measured as to how well it classifies the data and given a weight dependent on its error. Where a high weight is given to the training data if the error is good and low weight if bad.

One popular algorithm for boosting is AdaBoost, due to its simplicity, compared to other algorithms, its solid theoretical foundation, accurate predictions and wide use in different applications [Wu et al., 2007]. AdaBoost is an adaptive boosting algorithm, which means that the weights that are computed are updated in later rounds. Therefore the weights

placed on the data are exaggerated in later rounds as those weights are used again.
The AdaBoost algorithm, as presented by Freund and Schapire [1999], takes a training set as input in the form of $(x_1, y_1), ..., (x_m, y_m)$ where each x_i is in some domain or instance space X, and every y_i is in a label set Y. In terms of this project, this will be the positive and negative features calculated using ICF. AdaBoost then uses a weak or base learning algorithm across the input in rounds of $t = 1, ..., T$, where the algorithm maintains a set of weights over the training set, which is initially equal, however, over the number of rounds the algorithm concentrates on the hard examples in the training set. A weak hypothesis for X is formed by $h_t : X \to \{-1\} + 1$, where the goodness of the weak hypothesis is measure by its error

$$\epsilon_t = Pr_{i \sim dt}[h_t(x_i) \neq y_i] = \sum_{i:h_t(x_i) \neq y_i} D_t(i) \tag{6.11}$$

where the error is in relation to the distribution. Once the hypothesis has been formed a weight, α_t, is chosen based upon

$$\alpha_t = \frac{1}{2} ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) \tag{6.12}$$

where ϵ_t is the error or the hypothesis.

After which the distribution is updated for the next round by

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \tag{6.13}$$

where Z_t is a factor that normalises the data so that it is a distribution. The final step in the algorithm is to calculate the weighted majority vote of

$$H(x) = sign \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right) \tag{6.14}$$

where the *sign* can be used as a threshold for binary classification, where if the summation is below zero, the image is classified as a negative, and a summation above zero is a positive.

6.5 Classification

Now that the features calculated by ICF have been boosted using AdaBoost, these features can be used for classification. There has been a trade-off between accuracy of detection and speed of a classifier in computer vision. Where earlier iterations of detection algorithms applied a classification function and various positions and scales, were demanding in terms in computations. However, using the classification function as described by Bourdev and Brandt [2005], these issues can be avoided. Instead of using a standard cascade, which have multiple distinct cascade stages, their method is to decompose a larger classifier into a sequence of sub-classifiers, a term called a Soft Cascade. In this approach a cascade threshold can be set according to the values calculated by AdaBoost. The basic approach of a Soft Cascade as presented by Bourdev and Brandt [2005] can be comprised into two parts:

1. Instead of using a binary-valued approach, where either the given image is a sign or not, the stage is scalar-valued with the decision function being proportionate to how well this image passes the current stage and to the importance of that stage.

2. The decision function that determines if the given image passes a given stage depends on the values of the previous stages in the cascade, rather than only the current value.

Due to this approach the classifier can accrue information monotonically and it is possible to clearly see a difference between signs and non-signs. The accumulation of all previous stages in the cascade gives the Soft Cascade a function of:

$$H(x) = \sum_{t=1,\ldots,T} c_t(x) \tag{6.15}$$

where

$$c_t(x) = \alpha_t h_t(x) \tag{6.16}$$

are the set of weak classifiers and their weights determined by AdaBoost. The values of $h_t(x)$ as a function of t for a sample x, can be determined as a sample trace [Bourdev and Brandt, 2005]. The summation of the sample trace over a large number images shows a clear difference in the sample trace between positives and negatives. Bourdev and Brandt [2005] show a difference in the sample trace over 2,500 images of faces and non-faces in Figure 6.11.

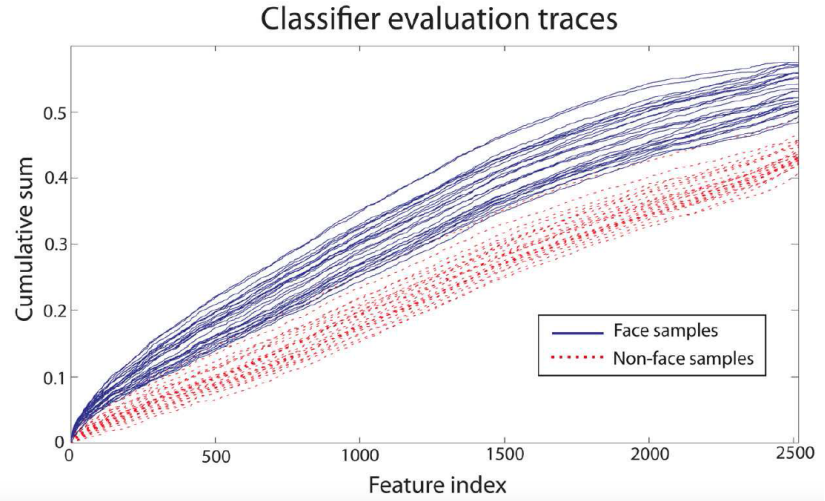


Figure 6.11: Sample Trace of 2500 face and non-face images [Bourdev and Brandt, 2005].

To determine if the given image is a speed sign or not, a set threshold of r_t is checked for each of the T features. If the cumulative sum is less than r_t, the given image is rejected as a speed sign. Whereas the images which are fully evaluated in the cascade structure are accepted as speed signs. The threshold r_t, is a negative value that can be altered according the required performance of the classifier, in terms of correct detections. One of the main advantages of a Soft Cascade is that it uses the information from previous stages of the classifier. This also means that it does not place too much emphasis on a given stage, as a positive image may fail a stage along the cascade but still be identified as a speed sign. This in turn means that stages of the cascade can be small and fast in terms of evaluation. [Bourdev and Brandt, 2005]

Chapter 7

Implementation

This chapter will include the overall implementation of the theories explained in the previous chapter. The algorithm is divided into two branches; a training algorithm and a detection algorithm. The training algorithm explains how the detector was implemented, after which the detection algorithm uses this trained classifier for detecting speed signs. The flow throughout the Chapter will be based upon the main algorithm which can be seen in Figure 6.2. The ICF approach was implemented using OpenCV, an open source computer vision library [OpenCV, 2015b], and using C++. Parts of the training algorithm was performed in MATLAB [MATLAB, 2015], namely boosting.

7.1 Training Algorithm

This section will explain the training algorithm, which will lead to a classifer that can determine whether or not a given image patch is of a traffic sign. The algorithm is a part of the main algorithm which is explained in Section 6.2, and is notated as *Train Boosted Classifier* in Figure 6.2.

The training algorithm can be seen in Figure 7.1. The algorithm runs offline from the rest of the program, due to the fact that it only needs to be run once. The result is a detector which is able detect given objects. The algorithm consists of four blocks, each will be described in this section.

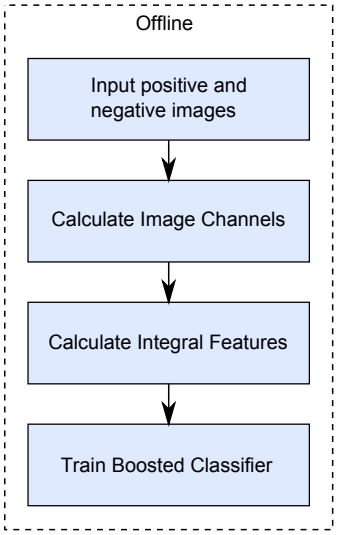


Figure 7.1: A flow-chart representing the training algorithm, of which the detector is trained.

7.1.1 Training Data-set

Training the detector requires a large amount of images. Two categories of images are required; one only containing the specific object, and another without the object, called positive and negative images respectively. In this project the German Traffic Sign Recognition Benchmark data-set was used due to the large number images available, of which there are 12,780 speed signs [Stallkamp et al., 2012]. These images are also in many different lifelike scenarios, such as dark and light conditions, which help in detecting speed signs in different contexts. Which was one of the requirements set out in Section 5.4. Examples of images from the data-set can be seen in Figure 7.2.

Figure 7.2: Examples of images from the German Traffic Sign Recognition Benchmark dataset [Stallkamp et al., 2012].

From the positive training set, 11,780 images of speed signs were used for training purposes, where the final 1,000 were saved for evaluation purposes. The implementation required the images to be of the same size, therefore all training images were resized to a size of 64x64 pixels. This size was chosen due to the original sizes of the data-set, where the images had a range of 30x30 to 90x90 pixels. To make a comprimise between these ranges a size in between these were chosen. The negative data-set consisted of 126,472 images, were created by taking patches of larger images, where the patches were the same size as the positive resized images. These images did not include traffic signs.

The following sections will describe the creation of channels and feature calculations performed to the images.

7.1.2 Calculate Channels

The *Calculate Image Channels* block in the training algorithm (Figure 7.1), can be extended to a flow-chart of the algorithm of that particular block. The flow-chart can be seen in Figure 7.3 and this section will explain the different elements.

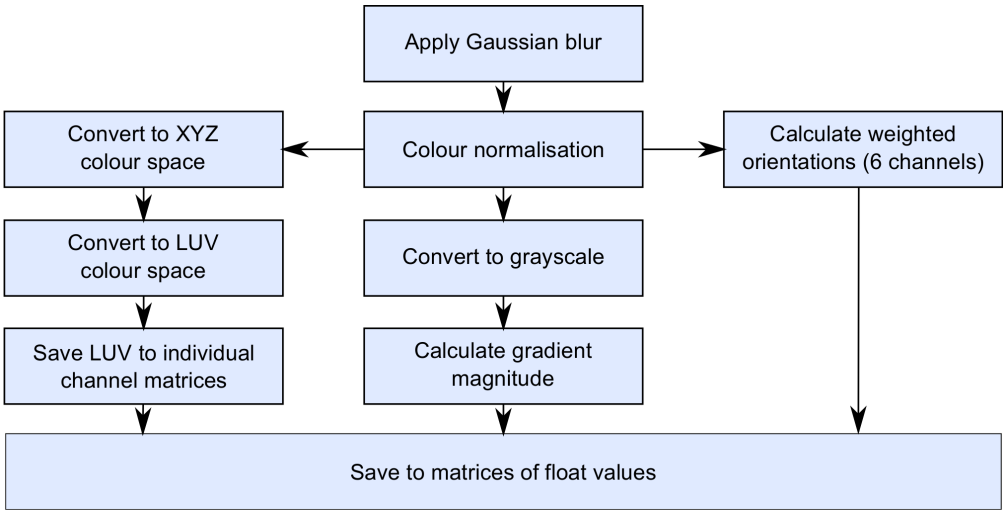


Figure 7.3: A flow-chart representing the algorithm for calculating the 10 channels; L, U, V, gradient magnitude, and the six different orientations.

Gaussian Blur

The input image is pre-smoothened in order to remove some noise. Dollár et al. [2009] states that post-smoothing is less useful compared to pre-smoothing, therefore a Gaussian filter is applied before the integral channels are computed. In a Gaussian filter the pixels are weighted, the center pixels has the highest weight, and the nearest neighbours have the second highest weight and so forth. Averaging based on the neighbourhood, causes pixels that vary a lot to get closer to the neighbourhood pixels. An example of this is salt-and-pepper noise, pixel that are either completely white or black that are of great difference to the rest of the image. Averaging this type of noise, causes these pixels to have less influence on the image. It results in a blurred image, and the amount of blur is defined by the size of the kernel. [Moeslund, 2012] In this project, a kernel size of 3x3 was used. Figure 7.4 shows an example of a training image before and after applying the Gaussian Filter.

(a) Input image. (b) Blurred input.

Figure 7.4: An example of an image before and after applying a 3x3 Gaussian kernel.

Colour Normalisation

After applying a Gaussian filter, the image is colour normalised. Situations may arise where the contrast is very low. For example, during dawn/dusk, or if the sun creates a silhouette around the sign. In these situations colour normalisation would add contrast to the images and would in turn help with the channel and feature calculations in latter parts of the system. Contrast Limited Adaptive Histogram Equalization (CLAHE) is often used for colour normalisation in object detection, due to the ability of distributing the contrast more evenly. CLAHE is an improved version of Adaptive Histogram Equalization (AHE), where contrast is evenly distributed, but noise is added in areas with nearly no contrast. The noise is removed in CLAHE by splitting the image into tiles, where histogram specification is used in each tile. The tiles are mapped, with regards to the neighbouring tiles, in order to remove noise. Peaks in the histograms are cut off, and evenly distributed in the images histogram, in order to maintain the same amount of pixel values. A visualisation of the cut-off can be seen in Figure 7.5. [Reza, 2004]

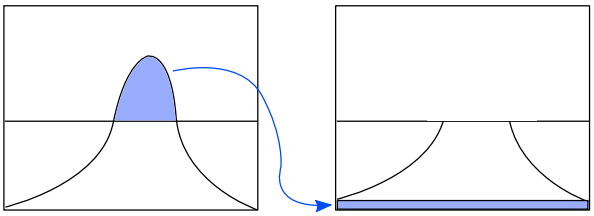


Figure 7.5: The visualisation of the cut-off in CLAHE. In the left histogram, the peak is cut-off, and distributed to the bottom of the right histogram.

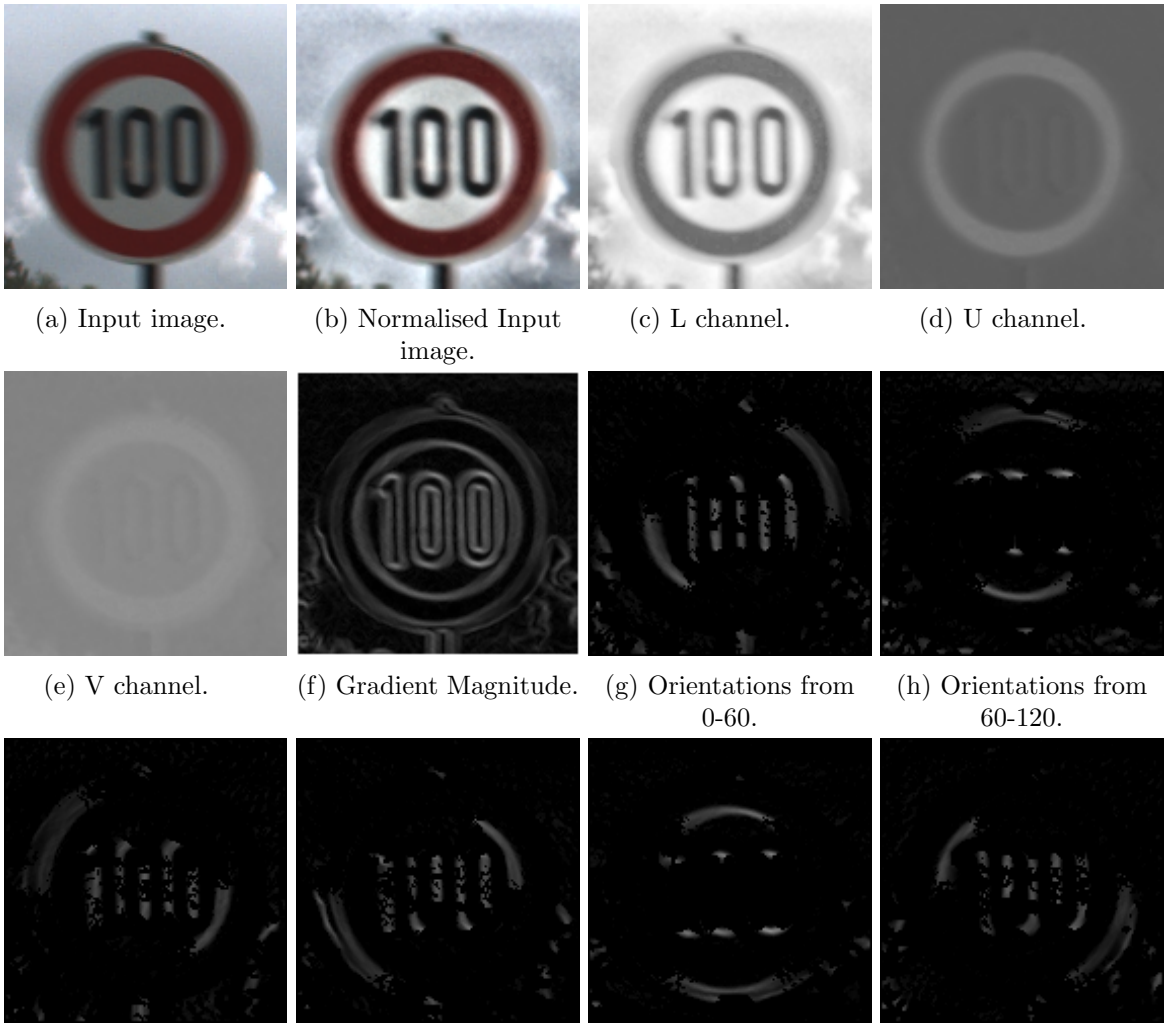
The cut-off is a constant limit that is defined based on the application. In this project a limit of 4 for the distribution parameter was used. The result is an image with higher contrast compared to the input image, and minimum noise compared to AHE. An example of two images, before and after CLAHE has been performed, can be seen in Figure 7.6.

(a) Input low contrast image. (b) CLAHE performed to image a. (c) Input low contrast image. (d) CLAHE performed to image c.

Figure 7.6: Two images before and after the colour normalisation CLAHE is performed.

Final Channels

Once the image has been smoothened and colour normalised, the channels can be computed. The LUV-channels are the first to be computed. The implementation is a function based on the theory described in Section 6.2.1. The output is a matrix with float values. Gradient magnitude is calculated with the theory described in Section 6.2.1. The gradient magnitude channel is then used to calculate the six different gradient histogram channels described in Section 6.2.1. The resulting channels can be seen in Figure 7.7 for a specific image from the positive training set.



(a) Input image. (b) Normalised Input image. (c) L channel. (d) U channel.

(e) V channel. (f) Gradient Magnitude. (g) Orientations from 0-60. (h) Orientations from 60-120.

(i) Orientations from 121-180. (j) Orientations from 181-240. (k) Orientations from 241-300. (l) Orientations from 301-360.

Figure 7.7: Illustration of 10 channels, and the input RGB image (a) and a colour normalised version (b).

Once all 10 channels are calculated, their values are normalised between 0-1. The program continues to the next block of the training algorithm; Calculate Integral Features.

7.1.3 Calculate Integral Features

After calculating the the image channels, the next step in the training algorithm is the *Calculate Integral Features* block. This block is extended to a larger flow-chart to illustrate the process of calculating the features of speed signs from the various channel computed. Figure 7.8 illustrates the extended flow chart.

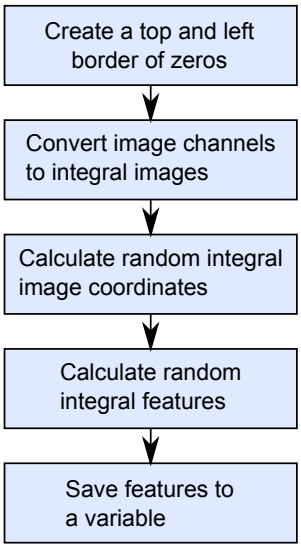


Figure 7.8: A flow-chart representing the algorithm for calculating integral features.

Based on the theory described in Section 6.3, first order features are the sum of pixels and can be calculated more efficient using integral images. Each of the 10 channels are converted to integral images. In order to do so, a border of zeros is added to each channel, to make sure that all pixels in the image can be converted without checking outside the image. An integral image can be efficiently set by a single pass over the image by

$$I(x,y) = i(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1). \tag{7.1}$$

Once the integral values for all pixels of the channels are computed, random features can be calculated. The positions of the features has to be consistent over all images, thus the positions for each feature is saved and from which channel it was computed. The area of these features should be at least 25 pixels [Dollár et al., 2009]. The feature position and channels are randomly calculated by a uniform distribution. The features can be calculated using the formula described in Section 6.3.

7.1.4 Boosting Features

Once the integral features have been calculated, the final stage of the training algorithm, *Train Boosted Classifier* can be done. This will be done based upon the theory presented in Section 6.4. The negative and positive sets of features calculated by the implementation from the previous section can be used as weak learners. AdaBoost uses these sets of features to create a strong learner, an algorithm that can identify if a given image is a sign or not. AdaBoost was implemented with Piotr's Computer Vision MATLAB Toolbox [Dollár, 2014]. The general program flow of AdaBoost implementation can be seen in Figure 7.9.

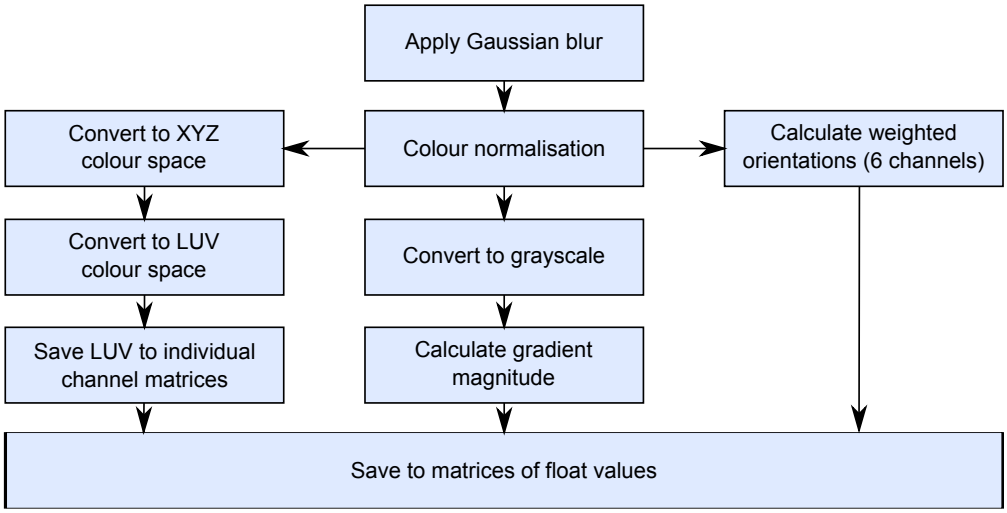


Figure 7.9: A flow-chart representing the implementation of AdaBoost.

The MATLAB function *adaBoostTrain* is run with the features calculated in Section 7.1.3 as inputs and with a number of AdaBoost options. These options consist of:

1. The number of weak classifers: [125 250 500 1000].
2. Decision depth tree length: depth-2.

The function runs the boosting algorithm over a number of rounds and from the error (explained in 6.4), adds weights to the selected hypotheses, which in this case are the specific features calculated. The MATLAB implementation differs from the theory explained in Section 6.4, in the way that the boosting algorithm uses decision-trees for binary classification rather than decision-stumps. In this case a depth-2 decision-tree is boosted, where a decision-tree takes a random weighted feature and checks if that value is above or below a threshold and classifies it accordingly. This is done in two steps. An example of a depth-2 decision tree can be seen in Figure 7.10.

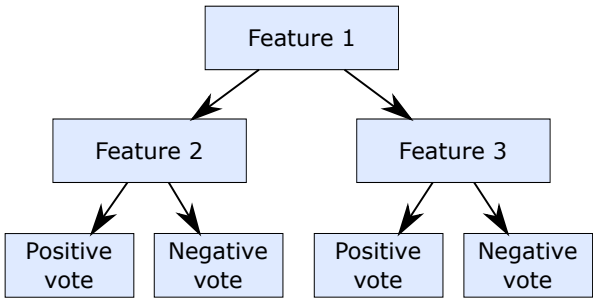


Figure 7.10: An example of a depth-2 decision tree.

The number of rounds in point one above, causes the weights on the earlier hypotheses to be exaggerated. Therefore the initial 125 hypotheses are boosting in four rounds, 250, in three, 500, in two and 1000, once. While the aim was to boost to 1000 features, the algorithm ended in the fourth round after 686 features, due to the algorithm converging to a point with a minimum amount of error, to which the strongest learner was found.

Once the *adaBoostTrain* function has ended, a classifier is outputted which can be further used with Dollárs toolbox. In order to use the classifier in the detection algorithm, which is in C++, parts of the classifier are saved to a file, namely the *feature IDs* selected by AdaBoost, their corresponding *feature thresholds*, and their *confidence values*.

7.2 Detection Algorithm

This section will explain the implementation of the detector. Each step required in order to input a large image, and detect speed signs in that given image, will be described. The section will follow the general flow throughout the detection algorithm shown in Figure 7.11. Some of the procedures explained in Section 7.1, will be used in this section as well as they same features that are boosted and chosen in AdaBoost are required for detection.

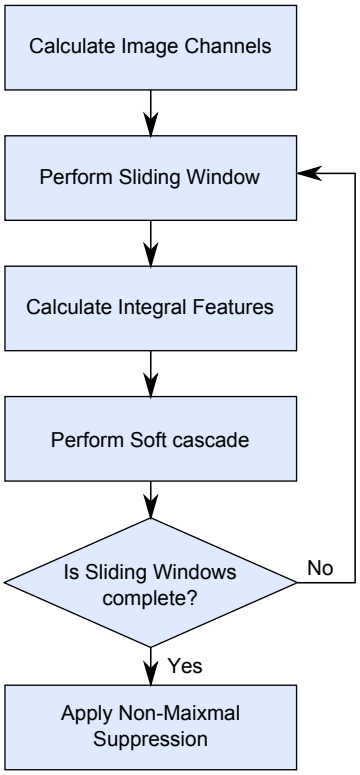


Figure 7.11: A flow-chart representing the detection algorithm, of which the general flow of the program is visualised.

7.2.1 Calculate Image Channels

Once an image is given as input, each of the channels described in Section 7.1.2 will be calculated on the image, and will be implemented using the same approach as described. However, instead of calculating the features on a single image, a larger image is used, with patches of 64x64 pixels, the same size as the classifier. The following section will describe the sliding window approach which divide the image into patches. For performance reason, integral versions of the channels will be calculated in this step.

7.2.2 Sliding Windows

This section will describe the block of the detection algorithm called *Sliding Windows*. A more precise representation of the sliding window-algorithm in form of a flow-chart can be seen in Figure 7.12. This section will explain the sliding window approach for detecting speed signs in a larger image.

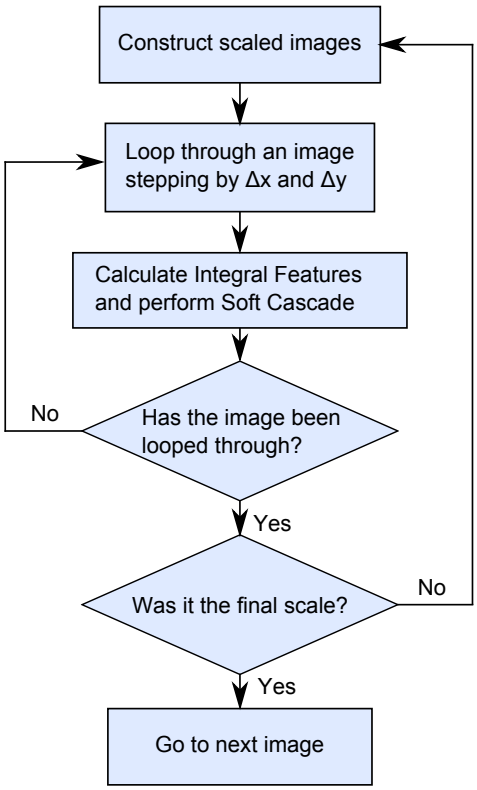


Figure 7.12: A flow-chart representing the sliding-window algorithm.

A sliding window is made by moving an image patch in pixel increments over the entire image [Forsyth and Ponce, 2002]. By sliding across the image, the trained classifier can be used to determine if the current patch is a speed sign or not. In this project an increment of 6 pixels was used. An illustration of this movement can be seen in Figure 7.14.

Figure 7.13: An example of a large image with multiple speed signs.

In each position of the image, a patch of the image is created. The patch has a size of 64x64, the same size as the boosted classifier, explained in Section 7.1.4. When searching for a traffic sign in a large image, the process of sliding through the window with a small stride, should ensure that the sign is included in an image patch. An example of a large image with speed signs can be seen in Figure 7.13. Also the sign may not be as the same size as the classifier, therefore different scales of the input image are created, and the sliding window approach runs through each of these images. By scaling the input image, the possible traffic sign will be scaled as well, and would likely fit the patch in at least one of the scales. In this implementation the image was scaled with 8 different values, ranging from 1.0 to 1.7, incrementing with 0.1 each time. These scales was chosen based on Dollár's implementation, with a small increase in order to have a higher range of detection distance.

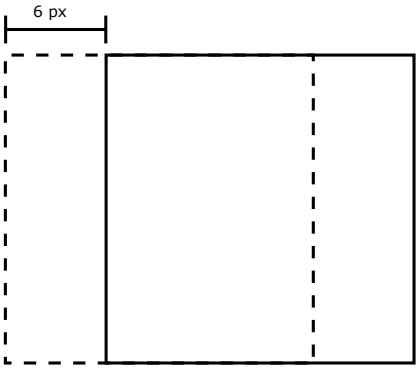


Figure 7.14: Illustration of the sliding window.

7.2.3 Calculate Integral Features

This section will describe the block of the detection algorithm called *Calculate Integral Features*. In each patch of the sliding window approach, the specific features calculated in Section 7.1.3 is required. It is important that the features calculated are in the same position and channel as the features boosted by AdaBoost, as explained in Section 7.9. The same approach for calculating these features is used as in Figure 7.8, with one minor change; the border of zeros is no longer needed, due to the fact that the patches are already integral versions.

7.2.4 Soft Cascade Implementation

The next part of the detection algorithm is to determine whether or not the features calculated in the indivual windows are from speed signs or not. This is done by using the values computed in AdaBoost from MATLAB. The classifier is used in every image patch, described in Section 7.2.2, and will detect if there is a speed sign in any of the patches.

The aim of the Soft Cascade is to update a decision function through summation. Where the updated function is checked if it has fallen below a cascade threshold. If it is below, the cascade ends and the given patch is denoted as a non-sign image. However, if the updated function is greater than the threshold, the Soft Cascade continues to the next set of decision trees. This process is shown in Figure 7.15.

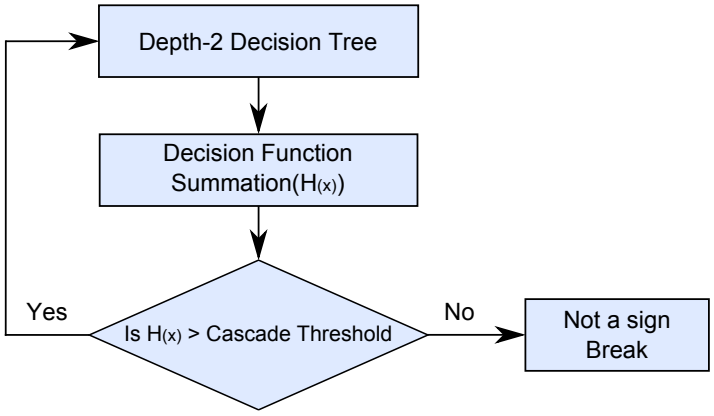


Figure 7.15: Flow-chart showing how the updated decision function in Soft Cascade is checked against a cascade threshold.

The decision trees use values from AdaBoost, which is built around the 686 weak classifiers. Three files that are imported, which are matrices of size 686x7, due to the fact that a single depth-2 weak classifier has seven potential nodes. The three files include the feature IDs, threshold IDs, and confidence values chosen by AdaBoost. The example of using a depth-2 decision tree with the seven potential nodes can be seen in Figure 7.16.

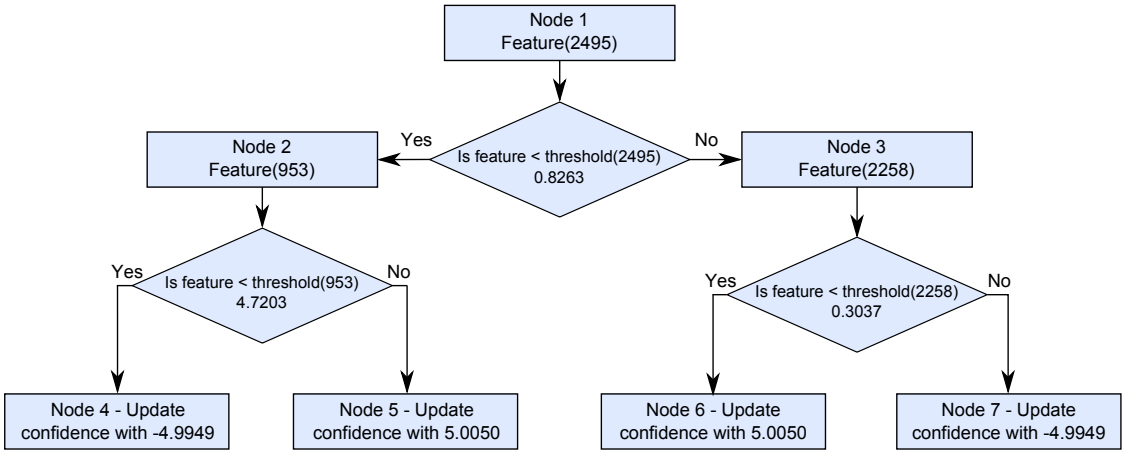


Figure 7.16: Flow-chart how a depth-2 decision tree updates the confidence value.

At each node, a specific feature chosen by AdaBoost is checked. In the figure above, feature number 2495 for the current patch is checked against a corresponding threshold, in this case if this feature is below 0.8263. If it is below, the decision tree continues to the second node, which is feature 953. This feature is then checked against its threshold (4.7203), if it is lower, the confidence is updated by -4.9949 in node 4, else by 5.0050 in node 5. By updating the confidence by a negative value, the decision tree indicates that the patch is not of a sign, whereas a positive value indicates a sign. The checks against the threshold in the first three nodes determine which of the final four the decision tree ends at.

Now that the confidence value has been computed in the decision tree, this can be updated in the decision function as shown in Figure 7.15.

7.2.5 Non-Maximal Suppression

The final block in the detection algorithm is to apply Non-Maximal Suppression (NMS) (see Figure 7.11). If the classifier works correctly it should detect numerous positives of speed signs after the Soft Cascade stage. This is due to the increments of 6 pixels, which is relatively small. Therefore the detector is able to detect one sign as many separate positives. This is a problem as it creates a large number of false positives. An example of multiple detections can be seen in Figure 7.17.

Figure 7.17: An example of a multiple detections of a speed sign.

The strategy for fixing this issue is to apply NMS to the positive detections found in an image after the sliding window approach is completed, this strategy suppresses nearby windows to a local maximum [Buil, 2011]. This is done by sorting the positive detections of speed signs according to their confidence value. After this, the list of the positives windows are checked in pairs, where if that pair overlaps by a chosen overlap threshold, between 0-100%. the window with the smaller confidence value is removed. This is repeated until all positive detections have been compared and determined that there is not a sufficient overlap between them. Once this is completed only separate classifications of speed signs remain. The final classifications should only be one instance of a given actual positive, however, NMS can also be used to suppress other false positives. If the detector finds a number of false positives in an area where there are no signs, these can be decreased to a single false positive. After applying NMS the multiple detections are suppressed to a single, this result can be seen in Figure 7.18.

Figure 7.18: Final result after NMS has suppressed the number of classifications to one.

Chapter 8

Detector Evaluation

In order to evaluate if the implemenation of a detection algorithm using ICF was satisfactory a number of evaluation methods are completed and presented in this chapter. This will be based upon the requirements set out in Section 5.5, namely: *Detect speed signs in a single image* and *Have a detection rate of 20%*.

8.1 Training Data-set Evaluation

Regarding the first point in 5.5, an initial evaluation of the training data can be made to see how well the detector finds speed signs of same form as the training set. This can be done by calculating the True Positive Rate, which is calculated by

$$\text{True Positive Rate (TPR)} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \tag{8.1}$$

The training set used to calculate the features in Section 7.1 from Stallkamp et al. [2012], consisted of a total of 12,780 speed signs, where 11,780 where used for training purposes, the final 1,000 signs were used for evaluation. As the evaluation set is roughly the same size as the training set, sliding window is not necessary, and the images are instead resized to the same size of the detector. The resulting TPR on the evaluation set at various cascade thresholds can be seen in Table 8.1.

-1	-2	-3	-4	-5
99.9%	99.9%	99.9%	99.9%	100.0%

Table 8.1: True positive rate on an evaluation set from Stallkamp et al. [2012].

However, it should be taken into consideration that the evaluation images are extracted from the dataset. To properly evaluate if the implemenation could detect a speed sign in a single image, a method should be used to determine how well the detector could detect speed signs in large images, as explained in Section 7.2. These will be presented in following sections.

8.2 Classifier Evaluation

The classifier can be tested to see the rate of which true positives occur, contra false positives. A visualisation of this ratio can be done with the Receiving Operating Characteristic (ROC) curve. ROC curves can be used to evaluate the trade-off between TPRs and FPRs (False Positive Rates). In the last decade it has been widely used to evaluate classifiers in machine learning and data mining research communities. ROC curves can therefore be used to visualise the classifier's performance. An example of an ROC curve can be seen in Figure 8.1.

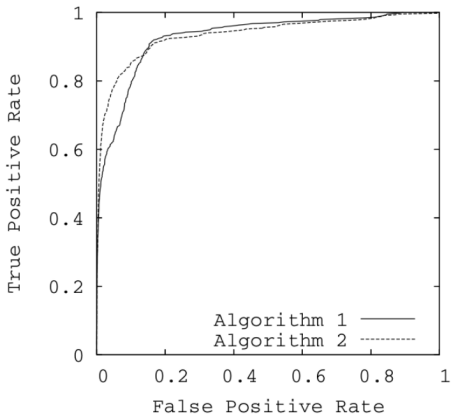


Figure 8.1: An example of a simple ROC curve, with true positive rate on the y-axis and false positive rate on the x-axis [Fawcett, 2003].

From these rates, it is possible to calculate precision, recall, and other parameters. Contrary to the TPR (see Equation 8.1), the FPR calculates how many true negative classifications are found, compared to the total number of negatives. This is calculated as:

$$\text{False Positive Rate (FPR)} = \frac{\text{False positives}}{\text{False positives} + \text{true negatives}} \tag{8.2}$$

In regards of object detection it would be relevant to see how many false positives the classifier finds per sliding window. This is calculated by

$$\text{False Positives Per Window (FPPW)} = \frac{\text{False positives}}{\text{Number of windows}} \tag{8.3}$$

The precision of the classifier, which is ratio between all true positive classifications and all positives, and can be calculated as:

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \tag{8.4}$$

Whereas, the recall rate of a classifier, which calculates true positives against total number of actual positives.

$$\text{Recall} = \frac{\text{True positives}}{\text{Actual positives}} \tag{8.5}$$

The most commonly used characteristics in an ROC curve are the TPR and FPR. Therefore the classifier in this project can be tested against an ROC curve with those characteristics.

The implementation of the boosted ICF classifier was evaluated using the German Traffic Sign Detection Benchmark (GTSDB), which comes from the same institute as the smaller training images used to calculate the original features [Houben et al., 2013]. From this dataset, 66 evaluation images, with a size of 1360x800 pixels, where in every image contained a speed sign. The signs varied in sizes, filling between 3,025 to 15,129 pixels. The calculation for the parameters of the ROC curve was calculated in a MATLAB script, using the boosted calculations from C++. Rather than using the Soft Cascade approach to classify windows, an adaptive threshold check was performed. For this all windows were processed through the entirety of the Soft Cascade, without discarding any images as non-signs. After this the confidence interval, minimum confidence interval, and XY-position were written to a file for the evaluation images. Now the Soft Cascade threshold can be altered after running the detector, to see what effect the threshold alteration would have on the detection rates. Before calculating these values, NMS was applied in the MATLAB script, in order to remove the false positives. The ROC curve with this adaptive cascade threshold, where the FPR and TPR are mapped, can be seen in Figure 8.2.

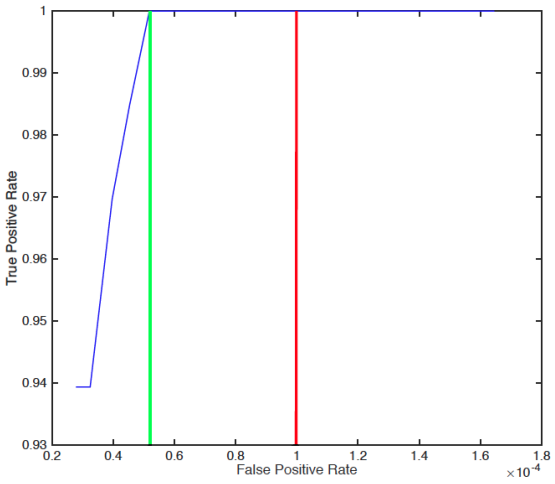


Figure 8.2: The ROC curve with the FPR on the x-axis and TPR on the y-axis.

The vertical red line in the figure represents Dollár's , goal FPR with, where they acheived a detection rate of 91.9%. At this FPR, the ICF implementation in this project has a TPR of 100%. This detection rate is reached at a lower FPPW than this, at $0.5181e^{-05}$, shown by the green line. However, these rates are not directly comparable as the evaluation for Dollár et al. [2009] was on a pedestrian dataset and not speed signs. Therefore the evaluation should be done on other detection implementations using the same data-set.

An issue with ROC curves when measuring performance of a detection algorithm, is that it can present a overly optimistic view, if there is a large skew in the data-set [Davis and Goadrich, 2006]. This is the case in the evaluation images for this implementation, as there are only 66 instances of actual positives with one sign in each image. However, there is a very large number of true negatives. This is due to the classifier implementation of 64x64 pixels and a sliding window step of 6 pixels. This results in a total of 8,067,708 windows across the 66 images, therefore actual negatives is equal to $8,067,708 - 66$. Which is a considerable skew in comparison to true positives.

Instead, a Precision-Recall (PR) curve can be used, which does not have this problem in evaluating algorithms. Precision shows how well the algorithm capture the total number of true positives with respect to true positives and false positives. Therefore if the implementation has a large number of negative examples, as is the case here, precision can show how this affects performance. On the hand, recall shows how the algorithm captures the total number of true positives with respect to actual number of positives. As in the ROC curve these two points are plotted according to various thresholds. The resulting PR curve can be seen in Figure 8.3.

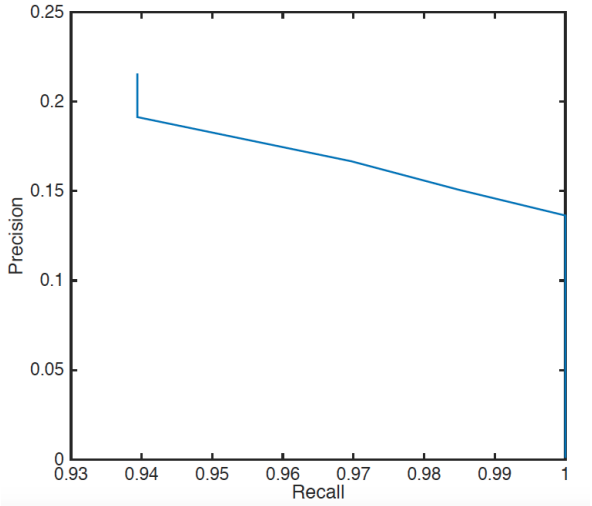


Figure 8.3: Precison Recall Curve.

The PR curve is not as favouring as the ROC curve. From the figure it can be clearly seen what effect the large skew has, with regards to negatves and positives. At the lowest threshold, the classifier has a high recall rate of 93.6%, where 62 of the 66 signs are identified. However, at that same threshold, 226 false positives are found, resulting in a precision rate of 21.5%.

It would also be beneficial to be able to compare various implementations and algorithms for traffic sign detection. One such method is to reduce the PR curve to a single scalar value that can represent the expected performance of the algorithm. This method is called Area Under the Curve (AUC), which calculates, as the name indicates, the area under the curve. With the AUC it is possible to compare different algorithms, where a higher value, indicating a greater area, shows better average performance. In this instance the AUC is calculated on the PR curve, resulting in a Precision-Recall Area Under the Curve (PR-AUC).

However, the AUC can not be computed from the PR values in Figure 8.3, as the PR curve must intersect both axes. These values cannot be computed as the lowest possible threshold in a Soft Cascade implementation. Therefore the curve must be interpolated to the point of Precision 1.0, Recall 0.0. A simple interpolation would however not give a proper representation of the PR-AUC, therefore a method from [Davis and Goadrich, 2006]. Instead of a linear interpolation, the local skew of the PR space is generated, which calculates how many negative examples are needed for each positive example. The local skew is calculated by:

$$ls = \frac{FP_B - FP_A}{TP_B - TP_A} \tag{8.6}$$

where A and B are the two points that are to be interpolated between. This can now be used to calculate new points for $TP_A + x$ to all integer values of x up to $TP_B - TP_A$. These integer values are then used to compute the new PR points by:

$$\left(\frac{TP_A + x}{TotalPositives}, \frac{TP_A + x}{TP_A + x + FP_A + ls} \right) \tag{8.7}$$

The resulting interpolation with the new PR curve can be seen in Figure 8.4.

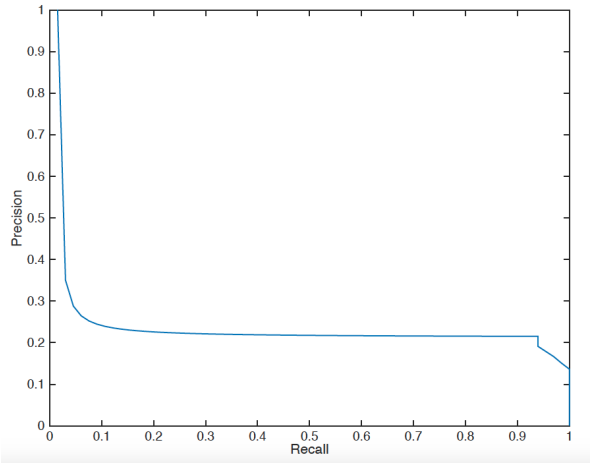


Figure 8.4: Interpolated Precison Recall Curve.

Table 8.2 shows the PR-AUC computed from the interpolated curve above, as well as the PR-AUC from other teams who have implemented ICF from the GTSDB data-set. Note that both *Visics* and *shawn_pan* have two seperate evaluations each, thus two entries in the table.

PR-AUC Results for ICF				
Visics	shawn_pan	Visics	shawn_pan	This Implementation
100%	99.99%	99.12%	88.35%	23.87%

Table 8.2: PR-AUC results for implementations of ICF on the GTSDB

8.3 Discussion and Conclusion

The evaluation results in the previous section gave both positive and negative results in terms of performance depending on which method was used. The requirements set out for the detector in Section 5.5, was that a detection rate of minimum 20% was needed, where an optimal detection rate should be above 80%. In terms of this, the detector performed very well when it was used to detect 1000 speed signs similar to the training set. At the lowest threshold of -1, 999 of the 1000 speed signs were detected. The detector acheived a recall rate of 100% when the threshold was set to -5.

However, the aim of this project is to implement an algorithm that can detect speed signs in a larger image and not just of the actual sign. The ROC curve also showed positive results. In comparison to Dollár et al. [2009], at a FPR point of 10^{-4}, they acheived a TPR of 91.9%, while this implementation had a TPR of 100% here. In fact, this TPR was acheived at a FPR of 0.5181×10^{-5}. While these results appear to be very positive and that the detection rate set out in 5.5 was achieved. There can be problems using an ROC curve to evaluate detectors when there is a large skew in datasets. This is the case, with only 66 actual positives in comparison to 8,067,642 actual negatives. Instead, using a precision-recall curve gives a better indication of performance as it takes false positives into account. The PR curves in Figures 8.3 and 8.4, show very promising recall rates, however, the precision rates are not as positive. Here a large number of false positives are found. Calculating the PR-AUC from the curve in Figure 8.4, results in a poor performance in terms of precision and recall in comparison to other implementations of ICF, as seen in Table 8.2. Such poor results is not optimal if this system should detect speed signs in a real-world environment.

There some parameters that could be altered in order to improve upon the number of true negatives and false positives. Firstly, the stride of the sliding window, as mentioned in Section 7.12, can be increased so that there is a decrease in the number of windows. The current implementation has a stride of 6 pixels, however altering this by one pixel has a very large effect, which can be seen in Table 8.3.

Pixel Stride	1	2	3	4	5
Number of Windows per Image	4415340	1103840	490593	275959	176614
Pixel Stride	6	7	8	9	10
Number of Windows per Image	122648	90109	68989	54510	44153

Table 8.3: The different number of windows per image according to the possible stride in the sliding window.

Here it can be seen that by simply increasing the stride by one pixel, the number of windows decrease by 26.5%, to 90,109 windows, and at a stride of 10 pixels the number of windows decrease with 64%. By decreasing the stride with a relatively small amount, the total number of true negatives drop significantly, which may result in a smaller number of false positives. While there is a chance that the recall rate decreases, a hypothesis could be made that the decreased rate in recall would be smaller in comparison to the improvement in precision, therefore resulting in an overall improvement in the PR curve and resulting PR-AUC.

Another method to decrease the number of potential false positives, is to alter the various scales that the original input image, as explained in 7.2.2. In the current implementation the input image is resized and the entire detection algorithm is run on a number of different scales. Currently, the width and height of the image is scaled by a factor between 1.0 and 1.7, with intervals of 0.1, resulting in 122,648 windows per image. Altering the scale changes the number of windows per image significantly. For example, using the same number of scales, but changing the interval from 0.1 to 0.2, reduces the number of windows per image to 88,138.

The number of windows can also be decreased by resizing the input image before running the current algorithm. The evaluation images were large, with a size of 1360x800 pixels. By resizing the input image, for example to a size of 640x480, the number of windows decrease by 71.1% to 35,494 windows. Future implementations could alter all three of these variables to constrain the number of false positives. Table 8.4 shows the number of windows per image on an input image resized to 640x480, also the interval scales are 0.2, and the effect of various strides can be seen.

Pixel Stride	1	2	3	4	5
Number of Windows per Image	921548	230387	102394	57596	36862
Pixel Stride	6	7	8	9	10
Number of Windows per Image	25599	18807	14399	11377	9215

Table 8.4: The different number of windows per image across various sliding window strides, the input images are resized and scales are increased.

This table shows that by keeping the stride the same, at 6 pixels, the number of windows decreases from 122,648 to 25,599. Increasing the stride in turn results in a much lower number of potential true negatives. However, by resizing the input image, the classifier training stage must be done again. As now potential signs in the images have decreased, the overall size of the classifier should be decreased from 64x64 pixels to for example 30x30 pixels. While altering these parameters will alter the number of true negatives, they should be changed with caution.

Other potential improvements to the detector, could be to use a combination of different channels, other than ones chosen in this implementation. This could for example be a grayscale channel, individual RGB channels, or other combinations. However, the 10 channels chosen for this project were based upon the positive detection rates achieved by Dollár et al. [2009].

A different approach could be taken in the final stage of the classification. Instead of a Soft Cascade, for example a Hard Cascade could be used. Here the output of the weighted hypotheses are not cumulative, but rather an image could fail a classification stage simply based upon a single learner. While this could present potential issues in actual positives being classified as false negatives, the potential number of false positives could also decrease.

Another approach of decreasing the amount of actual negatives could be to investigate context awareness. This approach eliminates areas of the image where the speed signs would not be present. An example could be if the hood of the car is shown in the bottom of the image, or if the sky makes up most of the top of the image. There it would not be relevant

to check for signs in the top and bottom of the image, since there would be no speed signs in these locations. Hence large areas of the image would not be processed, and therefore the amount of actual negatives would decrease, and overall speed is improved.

After completing the Soft Cascade implementation presented in Section 7.12, non-maximal suppression is used to decrease the number of positive classifications by removing classifications that overlap to the one with the highest confidence interval. The current implementation removes the classification with the lowest confidence if they overlap with 50%. This value could be altered to be more critical, for example stating that if two classifications overlap by 10%, the least confident one is removed.

Overall, the recall rate of the current implementation is very high, however, as mentioned, there are issues with precision, potentially due to the large skew. Therefore the parameters mentioned in this section should be analysed in future iterations of the implementation in order to improve the overall performance, especially in terms of precision.

Chapter 9

Interface Experiment

This chapter will present the interface experiment, which will be designed according to the project requirement specifications presented in Section 5.5. The main purpose of the experiment was to see if speed signs are missed by the driver, and how different modalities can be used to communicate this information. More specifically, the experiment lead to an evaluation of the three types of feedback, as well as general conclusions about driving behavior. Driving behavior will be evaluated through observations and electrodermal activity of the participants.

The experiment will be divided into several smaller experiment, each with different purposes. Each section will include the design the experiment, as well as the results and conclusions hereof. The experiments will be evaluated based on both qualitative and quantitative data.

9.1 Electrodermal Activity

Throughout the experiments, electrodermal activity (EDA) is logged using an Arduino UNO. EDA is used for describing electrical properties in the skin - the most used being skin conductance. This can be measured by applying an electrical component between two points of skin and measuring the flow between these two points. In the following experiments this is done by using copper foil and the resulting electrical flow is measured using an Arduino and logged at 50 Hz (see Figure 9.1).

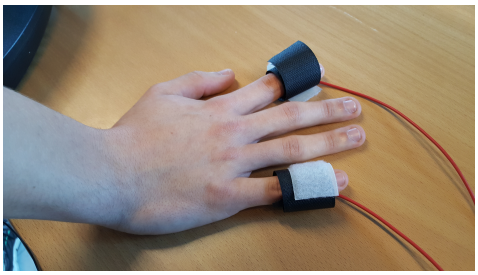


Figure 9.1: Example of how the EDA was logged through the Arduino.

The EDA of the participants can be very useful as it is a good indicator of arousal, which is bound to emotional and cognitive states. In this experiment, the EDA of the individual

test subjects will be used as an indicator to their processing while carrying out various tasks and as to their arousal when receiving the different types of feedback. In this case changes in their EDA can show subconscious states that may not be possible to measure otherwise. From the EDA, other measurements can be calculated, such as skin conductance level (SCL) and skin conductance response (SCR). SCL shows general changes in autonomic arousal over longer periods, whereas SCR can show changes in arousal in very short time spaces after an event. [Braithwaite et al., 2013]

9.2 Design of Feedback

This section will present a more elaborate discussion of the design of the feedback given to the driver, and takes a starting point in Sections 2.5, 4.2 and 5.5. In these sections it was stated that the most commonly used modalities in state of the art ADAS are visual, auditory, and haptics.

9.2.1 Visual

The visual modality can be very effective in presenting verbal-textual information, particularly when conveying longer messages. The visually presented information are usually in the form of graphics, text, or animations, and is a strong modality when conveying information representing real-world physical object, or information requiring persistent attention. To further enhance the comprehension of this information, graphics and textual descriptions can be used. [Stanney et al., 2004]

Relating this research to the context of this project, information about speed signs could be very well be communicated using the visual modality. As mentioned in Section 4.2, visuals are already used by some of the state of the art ADAS on the market, either in the form of head-up displays or on a display in the car.

With a focus on speed signs, the most straightforward way to display this information would be to show the exact visual representation of the speed sign corresponding to the given speed limit. Due to other factors in the test, however, it was decided not to display different variations of the visual feedback, and instead use a single way of representing that the driver is speeding. As a result, it was decided to use a warning sign. Furthermore, it was decided not to use a verbal-textual representation, since the duration it would take for the driver to comprehend the information would increase. Based on the Vienna Convention signs (Section 6.1), the feedback was created, and the final design of the visual feedback can be seen in Figure 9.2.

Figure 9.2: The visual feedback presented to the user in the test.

In the experiment, the visual feedback was given using Image Overlay Utility, which is a software that allows a visual overlay onto another running application [Image Overlay Utility, 2014]. The overlay was then presented on top of the driving simulator. The datalogger controlled the feedback by toggling whether it should be displayed or not. Section 9.6 will elaborate upon the setup in the experiment. Figure 9.3 shows how the overlay looked during the experiment.

Figure 9.3: Example of the screen overlay used in the experiment.

9.2.2 Audio

Unlike the visual modality, conveying information using audio is effective when the listener is either in motion and/or performing a task. Stanney et al. [2004] argue that the auditory modality is great for rapid communication of critical information such as for warnings and alerts, especially when dealing with time-relevant events and when immediate action is required. [Stanney et al., 2004]

When using the auditory modality, some general design guidelines should be taken into account. For example, the sound frequencies should be between 500 and 3,000 Hz, the duration of the sound should be longer than 500 ms, warnings and alerts should be simple and short, and in case of background noise the sound should be 15 dB above the threshold imposed by background noise. [Stanney et al., 2004]

It was decided to use a simple beeping sound as the auditory feedback. The sound was created using a Wav Tones, which is an online audio signal generator [Wav Tones, 2014]. With the above mentioned research in mind, the sound was designed with a frequency of 1,500 Hz, and a duration of 200 ms. This burst was repeated three times with a delay of 1,300 ms when presented in the experiment. In the experiment, the sound was played on a computer controlled by the data logger. A visualisation of the sound can be seen in Figure 9.4. The sound can be found on the attached DVD.

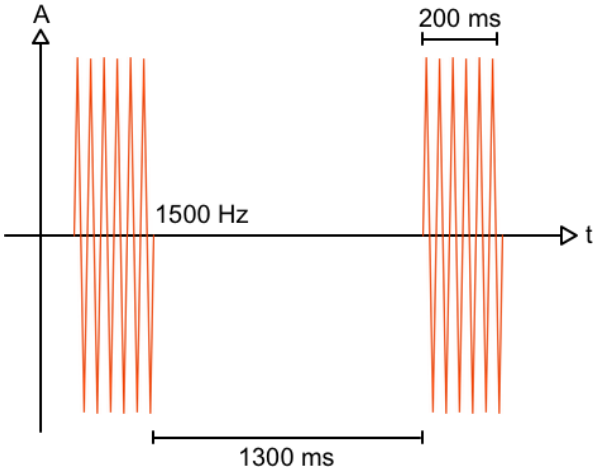


Figure 9.4: A conceptual visualisation of the auditory feedback presented to the user in the test.

9.2.3 Haptic

A haptic modality can work well when there should be a realtime alert. The modality could be placed in different areas of the car, however, it should be noted that the human body varies in sensitivity, where the limits of the perceptual threshold lies between 50 Hz and 600 Hz [Stanney et al., 2004]. Determining the level of the amplitude is dependent on what part of the body it is being used on. If the pattern was placed in the steering wheel the level can be relatively low, as the fingertips are quite sensitive. Here, the lower threshold of perceiving the vibration at 200 Hz is 0.07 μm [Jones and Sarter, 2008], therefore the amplitude should be above this so the pattern does not fall below this threshold when it decreases.

As the user is driving a car, it is important that the haptic feedback provides the information without being too disruptive. It was found that the ideal duration of tactile pulses should lie between 50 ms and 200 ms. [Jones and Sarter, 2008]

The haptic feedback alert can be designed to indicate caution by progressively decreasing the amplitude and using it as a cycle. The haptic feedback was implemented in the tests by using an Arduino UNO to control a small DC-motor with a weight attached. The intensity of the vibration is controlled by a pwm-signal, pre-programmed on the Arduino. The motor was attached to the steering wheel, such that the vibration ran through the entire wheel, and the participant would not be limited to place his/her hands at a specific position.

The vibration pattern consisted of six bursts, each with a 200 ms duration, followed by a 200 ms pause. The first burst had a duty-cycle of 78%, whereas the following bursts had a decrease of approximately 8%. A duty-cycle of 100% would be too disruptive, nor could it be below 30% since DC-motor would not have enough power to run. A visualization of the haptic pattern can be seen in Figure 9.5 below.

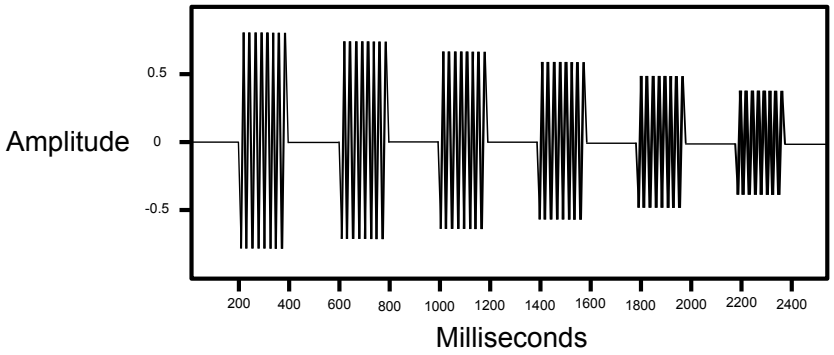


Figure 9.5: The pattern of the haptic feedback presented to the user in the test.

9.3 Setup

Unlike the field study presented in Section 2.6, these experiments were conducted using a driving simulator instead of an actual car. This was both due to practical reasons, and as a way to eliminate as many unwanted factors as possible throughout the experiment. The simulator used in the experiment was City Car Driving 1.4 [City Car Driving, 2015], which is a driving simulator designed for car driving education, with different virtual cities. The simulator has different features such as smart traffic AI systems, advanced car physics, and pedestrians - all of which can be adjusted accordingly. The routes driven in the experiment can be seen in Appendix B. To further add realism to the simulator, a Microsoft Sidewinder Force Feedback steering wheel was used to control the vehicle. As mentioned in the previous section, an Arduino UNO was used to control the haptic feedback, and another was used to log EDA data. All experiments were conducted in the same room with a setup as seen in Figure 9.6.

Figure 9.6: The physical setup used throughout the experiment.

Throughout the test, a facilitator and a datalogger were present. The role of the facilitator was to guide the participants through the test, by instructing and answering any questions that might arise. The datalogger was responsible for starting each simulation, as well as logging the EDA of the participants. Before conducting the actual experiments, the participants were introduced to the controls and drove freely to get accustomed to these. The following experiments were conducted on population of 12 participants. The gender of the participants was distributed as follows, 11 male and 1 female. All participants were university students, with ages varying between 21 and 25. As it was required to drive a car in a simulator and follow the traffic regulations, driver's license was required for all participants, furthermore their license should be acquired in a country following the Vienna Convention. Only one of the participants had experience with a driving simulator, before the experiment.

9.4 Feedback Intuitiveness

The purpose of this experiment was to get an idea about the intuitive understanding of the three modalities designed, while at the same time making it clear what the feedback actually meant in this experiment. The experiment aim to evaluate on the an item from the fourth point in the project outline; *Intuitive understanding of feeback* (see Section 5.5).

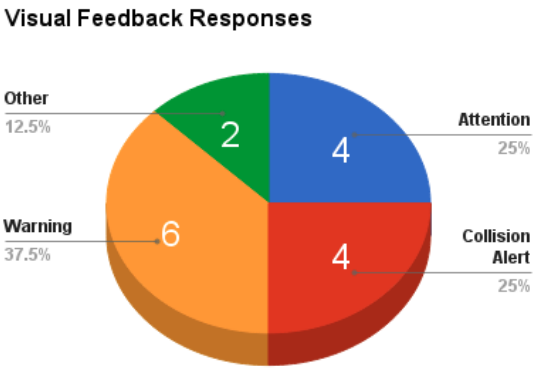
9.4.1 Design

The participants were placed in front of the display and steering wheel. They were be introduced to each of the three feedback modalities in a specific order. The order was visual, audio and haptic. After being introduced to each type of feedback the participants were asked to explain what they intuitively think the feedbacks indicate. The question which was asked after each feedback was: *In the context of driving a car, what is your intuitive understanding of this feedback?* After responding, the participants were told the meaning of the feedback in the test, which was that they were speeding. This was the case across all three types of feedback.

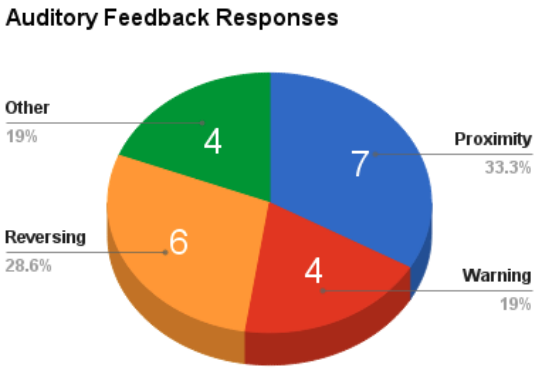
9.4.2 Results

The responses given by the participants were sorted into four categories. The categories consisted of the three most common responses, and a fourth category consisting of responses that did not fall into these categories. The responses can be seen in Figure 9.7 - note that the number of responses does not directly correlate with the number of participants, since some participants gave several responses to each type of feedback.

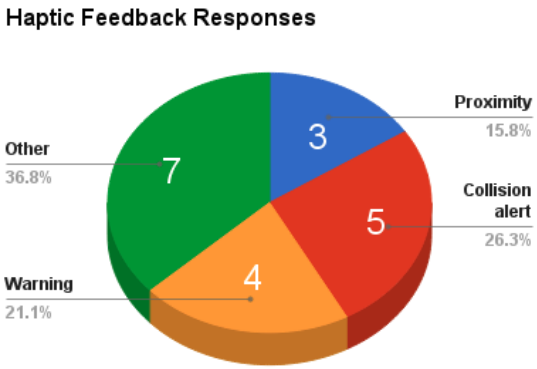
From Figure 9.7a, it can be seen that the almost all of the responses fell into the three main categories; *warning*, *attention*, and *collision alert*, whereas only two responses did not. After interviewing the participants, several suggested that the feedback should be more indicative, ie. a sign corresponding to the relevant speed limit, since this was the desired conveyed information.



(a) Categorised visual feedback responses.



(b) Categorised auditory feedback responses.



(c) Categorised haptic feedback responses.

Figure 9.7: Charts showing the responses to the question "In the context of driving a car, what is your intuitive understanding of this feedback?" for each modality.

In the post-test questionnaire, the participants were asked to rank the obtrusiveness of each type of feedback on a scale from 1-7. The mean values can be seen in Table 9.1. Based on these means, the visual feedback ranked the lowest (2.92), thus being the feedback rated least obtrusive by the participants. From the interviews, however, some argued that they gave a low ranking since they did not immediately notice the sign, which made them consider the feedback to be less obtrusive. Another comment which was made by several participants was that the sign was not eye-catching enough, and that it could be improved by adding contours, or by making it blink, or illuminate. It was also suggested to display both the current speed and the speed limit as visual feedback. Furthermore, the visual feedback was the most prefered of the three, as rated by the test subjects, whereas audio was the least prefered.

Most of the responses regarding the auditory feedback concerned *proximity* and *reversing*, with seven and six responses, respectively (see Figure 9.7b). The interviews showed that this was mainly due to the sound being recognisable to the participants, due to familarity and similarity to sounds currently implemented in various vehicle, such as parking/reversing sensors and reversing trucks. It was also suggested that using audio as a modality might not be preferred in a car since it could be confused with, or drown, in both interior and exterior noise. As per Table 9.1, the auditory feedback was rated 4.0 in obtrusiveness, thus being the most disturbing type of feedback.

	Visual	**Audio**	**Haptic**
Mean $\bar{x}$	2.92	4.0	3.0

Table 9.1: Mean ratings of the obtrusiveness of each type of feedback, based on user feedback on a scale from 1-7.

As seen in Figure 9.7c, the interpretation of the haptic feedback varied more compared to the other two modalities. The most common interpretations were *warnings* and *collision alerts*, but as it can be seen in the figure, seven responses fell in the *other* category. This category includes various different responses regarding e.g. road conditions and crashes. The mean rating of the obtrusiveness of the haptic feedback was 3.0. Several test subjects commented that they were not used to this type of feedback, which might have affected the variation in responses. Another repeated comment was that since the steering wheel is always an interface between the car and the driver, the haptic feedback could be used to let the driver know that something is up, and guide his/her attention. This could for example be towards the visual feedback, since this was sometimes overlooked. In general, many participants suggested to use multimodal feedback. Furthermore, as it was the case with the other modalities, it was suggested to make the feedback adapt according to the given situation.

9.4.3 Conclusions

From these results, several conclusions can be made. First of all, there was a widely different intuitive understanding of the three types of feedback, but some patterns occured. The majority of responses were concerned with warnings across all three modalities. This was most evident with the visual feedback, with a more illustrative representation of what the feedback meant. Visuals were also the preferred type of feedback from the three, while also be rated as the least obtrusive. The auditory feedback was mainly interpreted as a reversing/parking/proximity sensor, which was largely down to familiarity with the chosen sound. It was also ranked as the most obtrusive type of feedback. The interpretation of the haptic feedback varied more, however, it could be used to guide the driver's attention towards information presented elsewhere. Furthermore, in another iteration of the system, a multimodal approach should be implemented and tested as well.

9.5 Modality Reaction Time

The purpose of the second experiment was to evaluate the reaction time when the driver received feedback, as per the one of the items in point four of the project outline (see Section 5.5). The experiment will test how fast the participants react to each of the modalities. The hypotheses of this experiment were:

H: The drivers reaction time will differ depending on the feedback.
H_0: There is no difference between the reaction times of each type of feedback.
H_a: There is a difference between the reaction times of each type of feedback.

9.5.1 Design

The experiment was designed to minimize the number of disturbing factors in the simulation by disabling all AI elements, and by driving on an empty country road (see Figure 9.8. The test subjects were presented with each type of feedback once, in a counterbalanced order across all participants. Immediately after receiving the feedback, the participants should tap a marked area on the table with their left hand as fast as possible. The interval between the time the feedback was given and the area was touched was denoted as their reaction time. Furthermore, the SCR of the participants will be of interest, to investigate the arousal when receiving feedback.

Figure 9.8: Screenshot of the route driven in the reaction time experiment.

9.5.2 Results

The hypotheses are evaluated using analysis of variance (ANOVA) of the dependent variable that is the reaction time. ANOVA checks the difference in population mean values, $\bar{x}$, in all three samples, one sample for each type of feedback. This is done by analysing the variance of each sample. One of the participants are counted as an outlier and is therefore omitted from the test, resulting in a total of 11 observations, N, in each sample. The ANOVA gives a p-value of 0.0098, thus lower than the significance level of $\alpha = 0.05$. A p-value below 0.01 strongly suggest that there is a significant difference between one or more samples.

To evaluate which of the samples that differ significantly, a Tukeys HSD (Honest Significant Difference) test is performed. Tukeys HSD test shows that the difference between

visual- and auditory feedback, as well as between visual- and haptic feedback, is statistically insignificant, whereas the difference between auditory and haptic feedback are significantly different. Even though Tukey's HSD test did not result in a statistically significant difference between the samples, the sample means shows that the reaction time is shortest with the haptic feedback, and longest with the auditory feedback. The mean reaction times, as well as the samples' standard deviation, σ, can be seen in Table 9.2.

	Visual	**Audio**	**Haptic**
Obersvations, N	11	11	11
Mean, $\bar{x}$	1,678 ms	2,171 ms	1,245 ms
Standard deviation, σ	755.1 ms	766.2 ms	389.5 ms

Table 9.2: Mean reaction times and standard deviation for each type of feedback.

It was not possible to draw any conclusions from the SCR, since it became evident that the logged EDA data was not usefull. The reasons behind this, however, were unclear.

9.5.3 Conclusions

Even though the mean reaction times themselves give an indication of the differences across the three samples, a one-way ANOVA test was conducted. The dependent variable in the test was the meassured duration of the route driven. Based on the ANOVA's p-value of 0.0098, H_0 can be rejected and H_a accepted, thus giving the conclusion that there is a difference between the reaction times depending on feedback type. Performing a Tukey's HSD test, however, showed that there was only a statistically significant difference between auditory and haptic feedback. The experiment could preferably have been conducted on more participants, or been repeated by the participants for more observations.

9.6 Duration with/without Feedback

The purpose of this experiment is to evaluate the time it takes to drive a route, both with each type of feedback and without feedback. These durations will be compared to the time it takes to drive the route when complying with the speed limits. This experiment was designed to evaluate the fifth point in the project outline: *Observe general driving behaviour* (see Section 5.5). The hypotheses set in the experiment were:

H: The feedback used will affect the drivers completion time of the route.
H_0: There is no difference in the completion time of the route, between each type of feedback.
H_a: There is a difference in the completion time of the route, between each type of feedback.

9.6.1 Design

In this experiment the participants will drive a predefined route and receive feedback when they exceed the speed limits. They will be asked to drive according to their normal driving habits. The speed limit is 60 kph in the city unless other is advised through speed signs. A screenshot from the route can be seen in Figure 9.9. The participants will in this experiment be divided in four populations, one for each modality, and one who will not receive feedback

to act as a control population. This will be done in a counterbalanced manner. The feedback populations will receive feedback each time they exceed a speed limit for a period of two seconds.

Figure 9.9: Screenshot of the route driven in the feedback experiment.

9.6.2 Results

Like in the previous experiments, the results were evaluated using ANOVA, which resulted in a p-value of 0.209. The p-value in an ANOVA test is based on the F-ratio, which is calculated by dividing the variance between groups with the variance within groups. With an F-ratio close to 1.0, H_0 is true, and in this case $F = 1.78$. Thus, in this experiment H_0 is accepted and it cannot be concluded that there is a difference between the populations. The data gathered in the experiment can be seen in Table 9.3.

	Visual		**Audio**		**Haptic**		**None**		**Control***
	Time	Viol.	Time	Viol.	Time	Viol.	Time	Viol.	Time
1	4:42	4	5:10	2	4:30	3	3:39	19	4:52
2	4:10	7	5:03	3	3:44	4	4:42	7	4:40
3	4:43	2	4:31	3	5:00	11	3:37	17	4:48
Average	**4:32**	**4.3**	**4:55**	**2.7**	**4:25**	**6**	**4:03**	**14.7**	**4:47**

Table 9.3: Completion times in minutes and seconds, and violations throughout the route driven in the experiment 9.6, sorted according to population. *Control runs were driven by the group.

Table 9.3 also shows the number of traffic violations made by the driver during the route. It can be seen that the number of violations increased in the population who did not receive any feedback, compared to the ones that did receive feedback. This indicates that even though there were not significant differences in duration, the feedback still affected their driving somehow.

9.6.3 Conclusions

It was not possible to reject H_0, and thus it could not be concluded that there was a difference between the samples. As previously mentioned, the participants were divided into four populations. With 12 participants, this resulted in sample sizes of three. The low sample sizes are not optimal, and it could be preferable to conduct the experiment with more test subjects. Larger sample sizes might have affected the variances within- and between groups, thus resulting in a different F-ratio, and subsequently a lower p-value. However, it could be seen that there was a difference in the number of violations between the populations that recieved feedback and the ones that did not, indicating that the feedback did affect the driving positively.

9.7 Counting Signs

The purpose of this experiment was to investigate the effect of giving the driver an additional task while driving. As discussed in Sections 2.2 and 2.3, this could affect the driver attention, and increase their cognitive load. Furthermore, the participants' SCL should be affected. Like in the previous experiment, this also evaluates point five in the project requirement specifications (Section 5.5), namely to *observe general driving behavior*.

9.7.1 Design

As in the previous experiments, the driver was navigated throughout a pre-defined route, and again asked to drive according to their normal driving habits. This experiment differs in the way that the driver was asked to count all the road signs relevant to their driving throughout the route. An screenshot from the route can be seen in Figure 9.10 addition of this task should further increase the drivers cognitive load, which in turn might affect their driving behavior. The experiment will be evaluated mainly through observations and interviews, while also denoting the number of signs counted compared to the actual number of relevant signs on the route.

Figure 9.10: Screenshot of the route driven in the counting experiment.

9.7.2 Results

The experiment showed that the additional task did affect the driving. Through the interviews, it became apparant that the participants felt that the task was a hindrance to their driving. Most felt that it removed their focus from the road towards the signs, and that the counting itself became somewhat frustrating as the experiment progressed. On the other hand, one participant said that he could not always focus on the counting, since other factors in the simulator also required his attention. The majority, however, responded that they were more focused on counting the signs, than actually driving. This also becomes evident when looking at their driving performance. Figure 9.11 shows the different violations made throughout the experiment.

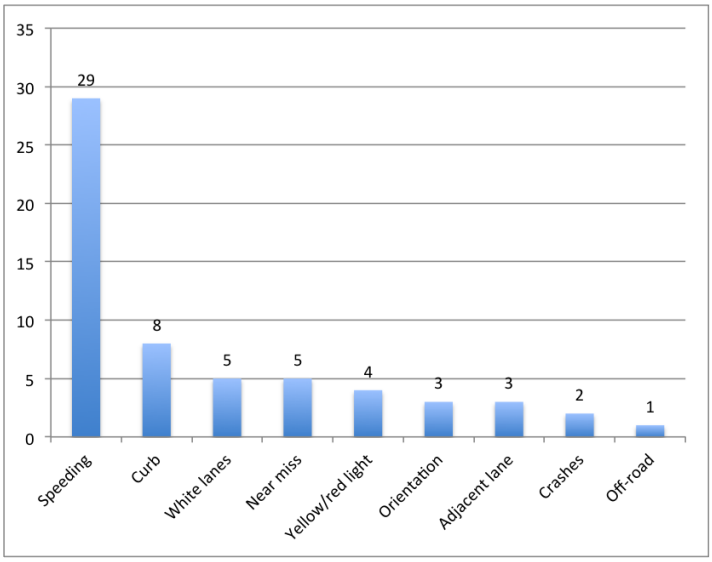


Figure 9.11: The most common violations seen throughout the experiment.

As seen in Figure 9.11, speeding was the most common violation, whereas lane departures (such as driving off-road, hitting the curb, driving on the lane markers, and driving in the adjacent lane) were also common occurances. Furthermore, several cases of driving through intersections with either yellow or red light were also seen, while several near misses and crashes also occured. Most of these violations were not as common in the other experiments, which might be related to the extra cognitive load that the task adds. Figure 9.13 shows screenshots with some of these violations.

Another interesting event during this test occured when the driver was briefly inattentive. As seen in Figure 9.12, the driver's attention was on the pedals for just a moment (Figure 9.12a), and at the same moment the car in front suddenly stopped (Figure 9.12b). As mentioned in the previous experiment, it was not possible to evaluate on the EDA data.

(a) Brief moment of driver inattention.

(b) Screenshot of the crash.

Figure 9.12: Inattention followed by a rear-end crash.

(a) Speeding 45kph in 20kph zone.

(b) Crossing a red light and near miss pedestrian.

(c) Lane departure.

(d) Near miss after right-hand turn.

Figure 9.13: Screenshots of some of the violations seen in the experiment.

9.7.3 Conclusions

By giving the drivers an additional task of counting signs while driving, their driving was affected. From both interviews and observations, it could be seen that the task partially moved their focus away from the road and more towards the signs. This resulted in numerous violations throughout the experiment. It would be interesting to test another population who drove the same route, but this time without the task.

9.8 Other results

In all experiments, the drivers were asked to drive as they usually would. When asked about what they focused on when trying to do so, many of the answers were the repeated by the test subjects. Figure 9.14 shows a graph of the amount of times a given answer was given. It can be seen that most answers were regarding elements in the surroundings, such as pedestrians, surrounding traffic, signs, and traffic lanes. Other responses concerned speed limits and controls (orientation and indicators).

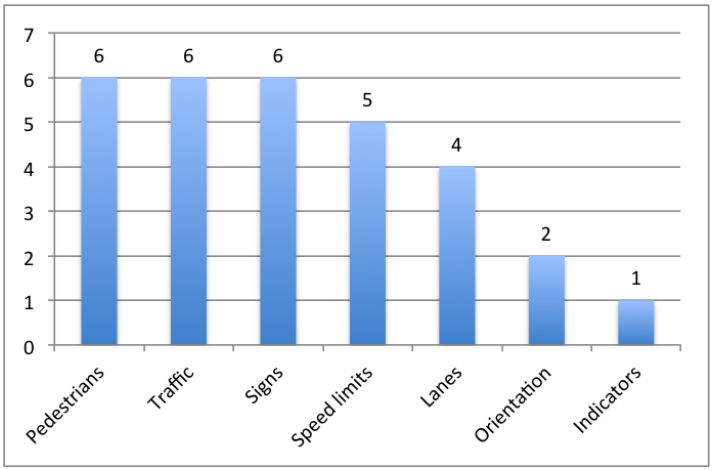


Figure 9.14: Responses to where the drivers' focus were when driving accordingly.

Finally, some participants commented on factors which were disturbing in the experiment. These were mainly regarding the controls, and the unfamiliarity of driving in a simulator. They said that the controls somewhat affected their driving, which is an important consideration regarding their results and feedback throughout the experiments. One test subject also commented that the fact that he knew this was an experiment, he was more focused on his driving and the traffic regulations, than when usually driving.

9.9 Overall Test Conclusions and Discussion

After conducting and evaluating each experiment, it is possible to draw some overall conclusions of the results, as well as discuss them. This will be done in this section.

The intuitiveness test showed a variety of different responses by the participants, when asked about the meaning of the feedback. Most responses, however, regarded alerts and warnings, especially with the visual feedback. With the purpose of informing the driver about speed limits, it might have been preferable to design the visual feedback as a sign corresponding to the given speed limit. This was also a frequent comment made by the participants during the interviews. Furthermore, it was suggested to make the feedback more eye-catching. In future experiments these aspects of re-design would definitely be points of interest.

The auditory feedback, was mostly considered as concerning parking/reversing. Based on the interviews, however, it became clear that this was almost entirely down to the participants being familiar with a similar sound, namely proximity sensors or reversing trucks. It was suggested to use a more descriptive auditory presentation of the information. This is somewhat contradicting to the research made, but would be an interesting aspect as well in further development.

The interpretation of the haptic feedback varied a lot, since the participants were not used to this type of feedback while driving. This was also one of the reasons to it being rated as the most obtrusive of the three. In future iterations of the system, the haptic feedback should be used in conjunction with one of the other modalities, as a way of guiding the drivers attention. In general, it would be interesting to create and evaluate a multimodal feedback system.

Comparing the reaction times of the different modalities showed that only the auditory and haptic feedback had a significant difference, even though the means indicated a difference in visual feedback as well. One way to improve on this data would have been to test the reaction times multiple times pr. participant, or, as an alternative, conduct the experiment on more participants.

Looking at the violations made while driving with a given type of feedback, indications could be seen that the feedback did affect the driving positively, but it was not possible to confirm this statistically. Larger sample sizes and more participants might have given better results. Violations were also of interest when adding the task of counting signs. The task did remove focus according to most participants. A consideration that should have been taken into account in the test design, however, was that no control group drove the route without counting signs. This would have made basis for a better comparison, and thus more concrete and concluding results.

As previously mentioned, EDA was logged throughout the experiment, however, the data was not satisfactory. This might be a result of the constructed equipment, which could have created noise. It would have been preferable to use other equipment to meassure the EDA. Alternatively, heart rate might have given indications of stress levels as well.

To summarise, the design of the feedback did not communicate the message of speeding flawlessly. It could probably benefit from re-designs and further evaluation, while a larger population would also be beneficial. Finally, a multimodal system might be a better solution, compared to communicating speeding information using a single modality.

Chapter 10

Discussion

Based upon the problem statement and the requirements set out in Sections 3 and 5.5, it can be determined that the detector implemented in this project was satisfactory. However, there is room for improvement in some areas, such as regarding the first requirement set out in Section 5.5, namely that the system should *Detect speed signs in a single image*. The first item of discussion for this point is the choice of computer vision method. While ICF may not be the current state of the art in terms of machine learning detection, it was chosen as an introduction to detection algorithms. ICF was implemented to a satisfactory degree where it was possible to detect speed signs. In future work, the possibility to classify the detected speed signs would be relevant. While the implementation of ICF was satisfactory, as mentioned in Section 4.3, there are other detection methods than machine learning which could be implemented, such as colour-based and shape-based detection. However, there are more constraints to these methods as colour-based methods can be difficult to implement due to varying degrees in contrast in driving situations. Shape-based detections, on the other hand, can have difficulties as speed signs could be viewed from different angles. These issues are taken into account with a machine learning method, however, only if there is access to a positive dataset with a large number of samples, where speed signs are present in varying situations, such as different weather conditions and different viewing angles.

The next requirement set out in Section 5.5, was regarding the detection rate of the system, where it was stated that it should *Run a detection benchmark*, with a detection rate of 20%. This required detection rate was set due to the system running at 5fps, where it was determined that the system should be able to detect a speed sign in at least one of the five frames. The detection rates shown in Chapter 8, were considerably above this rate, with the detector achieving a true positive rate of 100% at 0.5181×10^{-5} FPPW. However, detection rates are not the only way in which a detector should be evaluated, as a PR-curve and a PR-AUC show that the detector was quite poor in terms of precision, due to the large number of negatives. As mentioned, potential fixes to this could be made by altering either the sliding window stride, scales of the images, resizing the input images, or a combination of the previous three points.

There are other ways in which the overall performance of the system could be improved, for example with context awareness. Here, the system could attempt to determine where in

the image there could be a speed sign. For example it may be able to determine that a large dark area in front of the car is the road, and therefore the system would know not to search for speed signs here. It could also be determined that speed signs are only present on the right side of the image, as this is where most speed signs are located in when driving. Adding such context awareness this to the system would also help to improve on the current issues in terms of precision and the large number of actual negatives.

The current implementation is also not feasible regarding the requirement that *The detection of speed signs needs to run with a minimum of 5 frames per second.* The current implementation runs considerably slower than this. Currently, the processing time of an image is approximately 18 minutes. For future work this could be greatly improved upon, since it does not meet the functional requirements of 5 frames per second. Many of the aspects mentioned earlier to improve upon precision may also help improve the overall speed, as the detector would run on a smaller number of windows.

Where the computer vision implementation covered points 1 and 2 in the project requirement specifications, the interface experiment covered the remaining points. These requirements stated that the system should *Present the driver with visual, auditory, and haptic feedback* and that it should be possible to *evaluate the interface.* Thus, one of the most important aspects of the experiment was the design of the feedback, as presented in Section 9.2. Even though the feedback was designed based on research, compromises still had to be made in order to eliminate unwanted factors in the experiment. It became apparent that the participants understanding of the feedback varied a lot, which might be partially related to the design compromises. Some considerations in the design process were even suggested by the participants in the interviews. The visuals could have been more eye-catching, the audio more descriptive, and the haptic feedback could have been given through another interface or with a different pattern. Another thing that arose from the interviews was the repeated suggestion of designing a multimodal system. A multimodal system could utilise the strengths of each modality in a combined interface.

Another requirement in Section 5.5, stated to *Observe general driving behavior.* This was partially done through the number of the traffic violations made by the drivers. By giving the driver feedback whenever violating the speed regulation, he became more attentive towards his driving. As a result, the number of violations decreased, and a large difference could be seen in the number of violations depending on whether feedback was received or not. The feedback was sometimes considered distracting as well, and as mentioned in Section 2.3, inattention, internal-, and external distractions are some of the main factors behind the high amount of fatal traffic accidents. The feedback should inform and guide attention, rather than disturbing the driver.

The setup of the simulator itself (see Section 9.3) might also have affected the results. It became clear that the unfamiliar controls and setup affected the driving behavior, regarding both sensitive controls, virtual mirrors, and lack of physical orientation. While it would be preferable to conduct the experiment in an actual car, it could also be beneficial to have a more extensive simulator. It would be interesting to conduct the experiment with an eye-tracker to track where the driver's visual attention is when driving.

Chapter 11

Conclusion

The project took a starting point in the number of deaths and accidents in traffic accidents in Denmark. Where in 2013 there were 2,954 deaths and 51,063 accidents. The Danish Traffic Safety Commision have set a goal to reduce these figures by 50% by 2020. They outlined specific strategies, for example advancements in vehicle technology and ADAS.
From here the factors behind these statistics were researched, where some of the main culprits were problems with driver selective attention, distraction, and inattention. This can lead to driver overlooking or ignoring road information.
In order to get a first hand impression of driver habits, a driving experiment with a small sample was conducted, where many of the same items researched in the problem analysis were observed. On the basis of this, the aim of this project was to research how to implement a system that could detect road information and convey this to the driver of a vehicle, thereby reducing the number of accidents. This lead to the following problem statement:

> *How can computer vision be implemented in an advanced driver assistance system to recognise traffic sign information, and how can this information be unobtrusively conveyed to the driver?*

State of the art research showed that visual, auditory, and haptic forms of feedback could be used as modalities in an ADAS interface. However, when designing these, one should be wary of issues with distraction and complexity, which also became evident from the interface experiment conducted. Most participants commented on the visual feedback as a warning or call for attention, thus being the feedback leaning mostly towards speed warnings. The reaction time, however, was best with haptic feedback. Based on this, a multimodal system would be preferable, with haptics used as a direct interface to the driver which make the driver attentive towards visual feedback. The visual feedback could then be redesigned to be more representative of a speed sign.

From three different types of computer vision detection methods, it was chosen to take a learning-based approach, using ICF to calculate candidate features for detecting speed signs. Boosting these features with AdaBoost, it became possible to use a Soft Cascade classification technique to distinguish between sign and non-sign images. This algorithm proved to work very well when detecting speed signs, however, not as well in terms of precision due to the high number of false positives per window. Future iterations should aim to decrease the total

number of actual negatives and thereby reduce false positive classifications.

It can be concluded that the project requirements were fulfilled, both in terms of the detection of speed signs and evaluating a potential interface. Speed sign detection, as a minimum, was given an aim to have a detection rate of 20%. With a sliding window approach, it was able obtain a satisfactory recall rate of 93.6%, however, as mentioned the precision was poor. The interface experiment also gave indications into the use of modalities, where a multimodal system would be preferable. From the conclusions made, it can be determined that this project can be used as a starting point in working towards a larger system, as presented in the functional requirements.

Bibliography

ABIResearch (2015). Mercedes-Benz, Volvo, and BMW Continue to Dominate ADAS Market. https://www.abiresearch.com/press/mercedes-benz-volvo-and-bmw-continue-to-dominate-a/. [Online; accessed 17-March-2015].

BMW (2015a). BMW Technology Guide: Head-Up Display. http://www.bmw.com/com/en/insights/technology/technology_guide/articles/head_up_display.html. [Online; accessed 18-March-2015].

BMW (2015b). BMW Technology Guide: iDrive. http://www.bmw.com/com/en/insights/technology/technology_guide/articles/idrive.html. [Online; accessed 18-March-2015].

Bourdev, L. and Brandt, J. (2005). Robust object detection via soft cascade. *CVPR '05 Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2(2):236–243.

Braithwaite, J., Watson, D., Jones, R., and Rowe, M. (2013). *A Guide for Analysing Electrodermal Activity (EDA) and Skin Conductance Responses (SCRs) for Psychological Experiments*. Selective Attention and Awareness Laboratory (SAAL) Behavioural Brain Sciences Centre University of Birmingham.

Brkić, K. (ND). *An Overview of Traffic Sign Detection Methods*. Departmentof Electronics, Microelectronics, Computer and Intelligent Systems Faculty of Electrical Engineering and Computing Unska.

Brookhuis, K., Waard, D., and Jannsen, W. (1993). The use of psychophysiology to assess driver status. *European Journal of Transport and Infrastructure Research*, 36:1099–1110.

Brookhuis, K., Waard, D., and Jannsen, W. (2001). Behavioural impacts of Advanced Driver Assistance Systemsan overview. *European Journal of Transport and Infrastructure Research*, 1:245–253.

Buil, M. (2011). Non-maxima suppresion. http://academica-e.unavarra.es/bitstream/handle/2454/4626/577667.pdf?sequence=1. [Online; accessed 15-May-2015].

Bunn, T., Slavova, S., Struttmann, T., and Browning, S. (2005). *Sleepiness/fatigue and distraction/inattention as factors for fatal versus nonfatal commercial motor vehicle driver injuries*. Elsevier.

City Car Driving (2015). City car driving. `http://citycardriving.com`. [Online; accessed 14-March-2015].

Davis, J. and Goadrich, M. (2006). The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240.

Dollár, P. (2014). Piotr's Computer Vision Matlab Toolbox (PMT). `http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html`.

Dollár, P., Appel, R., Belongie, S., and Perona, P. (2014). Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Learning*, 36(8).

Dollár, P., Tu, Z., Perona, P., and Belonge, S. (2009). Integral channel features. `http://pages.ucsd.edu/~ztu/publication/dollarBMVC09ChnFtrs_0.pdf`. [Online; accessed 15-February-2015].

DTSC (2013). *Every Accident is one too many - a Shared Responsibility. Danish Road Safety Commission National Action Plan 2013-2020*. The Danish Traffic Safety Commission.

Fawcett, T. (2003). ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. *Intelligent Enterprise Technologies Laboratory*.

Forsyth, D. and Ponce, J. (2002). *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference.

Freund, Y. and Schapire, R. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780.

Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., and Igel, C. (2013). Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, pages 1–8.

Image Overlay Utility (2014). Image Overlay Utility. `http://imageoverlayutility.bitbucket.org`. [Online; accessed 10-May-2015].

Jansen, M. (ND). Unobtrusive interfaces: Preventing information overload in ambient interfaces. `http://hmi.ewi.utwente.nl/verslagen/capita-selecta/CS-Jansen-Michel.pdf`. [Online; accessed 8-March-2015].

Jones, L. and Sarter, N. (2008). Tactile Displays: Guidance for Their Design and Application. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 50:90–111.

Karray, F., Alemzadeh, M., Saleh, J., and Arab, M. (2008). Human-computer interaction: Overview on state of the art. *International Journal on Smart Sensing and Intelligent Systems*, 1(1).

Kovačić, K., Ivanjko, E., and Gold, H. (2013). Computer vision systems in road vehicles: A review. *Proceedings of the Croation Computer Vision Workshop, Year 1*.

Kun, J. (2015). Weak Learning, Boosting, and the AdaBoost Algorithm. `http://jeremykun.com/2015/05/18/boosting-census/`. [Online; accessed 19-May-2015].

Luoma, J. (1991). Perception of highway traffic signs: interaction of eye fixations, recalls and reactions. *Vision in vehicles: III*.

MATLAB (2015). Matlab. `http://se.mathworks.com/products/matlab`. [Online; accessed 1-February-2015].

Matthews, T., Dey, A., Mankoff, J., Carter, S., and Rattenbury, T. (2004). A toolkit for managing user attention in peripheral displays. *UIST '04: Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*.

Meng, F. and Spence, C. (2015). Tactile warning signals for in-vehicle systems. *Accident Analysis and Prevention*, 75:333–346.

Mercedes (2015). Mercedes Safety: Attention Assist, Pre-Safe and Distronic Plus. `http://www.mbusa.com/mercedes/benz/safety`. [Online; accessed 18-March-2015].

Moeslund, T. B. (2012). *Introduction to Video and Image Processing*. Springer London.

Møgelmose, A., Trivedi, M., and Moeslund, T. (2012). Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, 13.

NHTSA (2008). *National Motor Vehicle Crash Causation Survey*. US department of transportation, National Highway Traffic Safety Administration.

Nilsson, L. and Alm, H. (1991). Collision Avoidance Systems -Effects of different levels of task allocation on driver behaviour. *The Netherlands: University of Groningen, Traffic Research Centre*.

OpenCV (2015a). Miscellaneous image transformations. `http://docs.opencv.org/modules/imgproc/doc/miscellaneous_transformations.html`. [Online; accessed 3-April-2015].

OpenCV (2015b). Opencv. `http://www.opencv.org`. [Online; accessed 20-February-2015].

Oregon Institute of Technology (ND). Information on the CIE LUV Color Space. `http://dcssrv1.oit.uci.edu/~wiedeman/cspace/me/infoluv.html`. [Online; accessed 2-April-2015].

Porathe, T. and Strand, L. (2011). Which sign is more visible? measuring the visibility of traffic signs through the conspicuity index method. *European Transport Research Review: 3*.

Reza, A. (2004). Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement. *VLSI Signal Processing*, 38:35–44.

Schwiegerling, J. (2004). *Field Guide to Visual and Ophthalmic Optics.* SPIES Press, Bellingham, WA.

Shinar, D. (2007a). *Distraction and Inattention.* Traffic Safety and Human Behavior. Elsevier.

Shinar, D. (2007b). *Driver Infromation Processing: Attention, Perception, Reaction Time and Comprehension.* Traffic Safety and Human Behavior. Elsevier.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, (0).

Stanney, K., Sammen, S., Reeves, L., Hale, K., Buff, W., Bowers, C., Goldiez, B., Nicholson, D., and Lackey, S. (2004). A Paradigm Shift in Interactive Computing: Deriving Multimodal Design Principles from Behavioral and Neurological Foundations. *International Journal of Human-Computer Interaction*, 17:229–257.

Stutts, J., Feaganes, J., Rodgman, E., Hamlett, C., Meadows, T., Reinfurt, D., Gish, K., Mercandante, M., and Staplin, L. (2003). *Traffic Safety and Human Behavior.* AAA Foundation for Traffic Safety.

Tran, C., Doshi, A., and Trivedi, M. (2012). Investigating pedal errors and multi-modal effects: Driving testbed development and experimental analysis. *IEEE Conference on Intelligent Transportation Systems.*

UNECE (1968). *Convention on Road Signs and Signals.* United Nations Economic Commision for Europe.

United Nations (2015). UNTC. `https://treaties.un.org/Pages/ViewDetailsIII.aspx?src=TREATY&mtdsg_no=XI-B-20&chapter=11&Temp=mtdsg3&lang=en`. [Online; accessed 24-April-2015].

Vejdirektoratet (2014). Dødsulykker 2013, Årsrapport 2013. `http://www.vejdirektoratet.dk/DA/viden_og_data/publikationer/Lists/Publikationer/Attachments/836/DUS%202013.pdf`. [Online; accessed 27-February-2015].

Viola, P. and Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:511–518.

Viola, P. and Jones, M. (2012). Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154.

Volvo (2015). IntelliSafe — Volvo Cars. `http://www.volvocars.com/us/about/our-innovations/intellisafe`. [Online; accessed 18-March-2015].

Wang, G., Ren, G., Jiang, L., and Quan, T. (2013). Hole-based traffic sign detection method for traffic signs with red rim. *The Visual Computer*, 30.

Wav Tones (2014). Professional Online Audio Frequency Signal Generator. `http://www.wavtones.com/functiongenerator.php`. [Online; accessed 30-May-2015].

World Health Organisation (2004). *WorldReport on Road Trafic Injury Prevention*. World Health Organization, Geneva, Switzerland.

Wu, X., Kumar, V., Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G., Ng, A., Lui, B., Uy, P., Zhou, Z., Steinbach, M., Hand, D., and Steinberg, D. (2007). Top 10 algorithms in data mining. *Knowledge and Information System.*, 14:1–37.

Appendix A

Field Research Route

Figure A.1 below shows the route driven by the participants in the field research, as presented in Section 2.6.

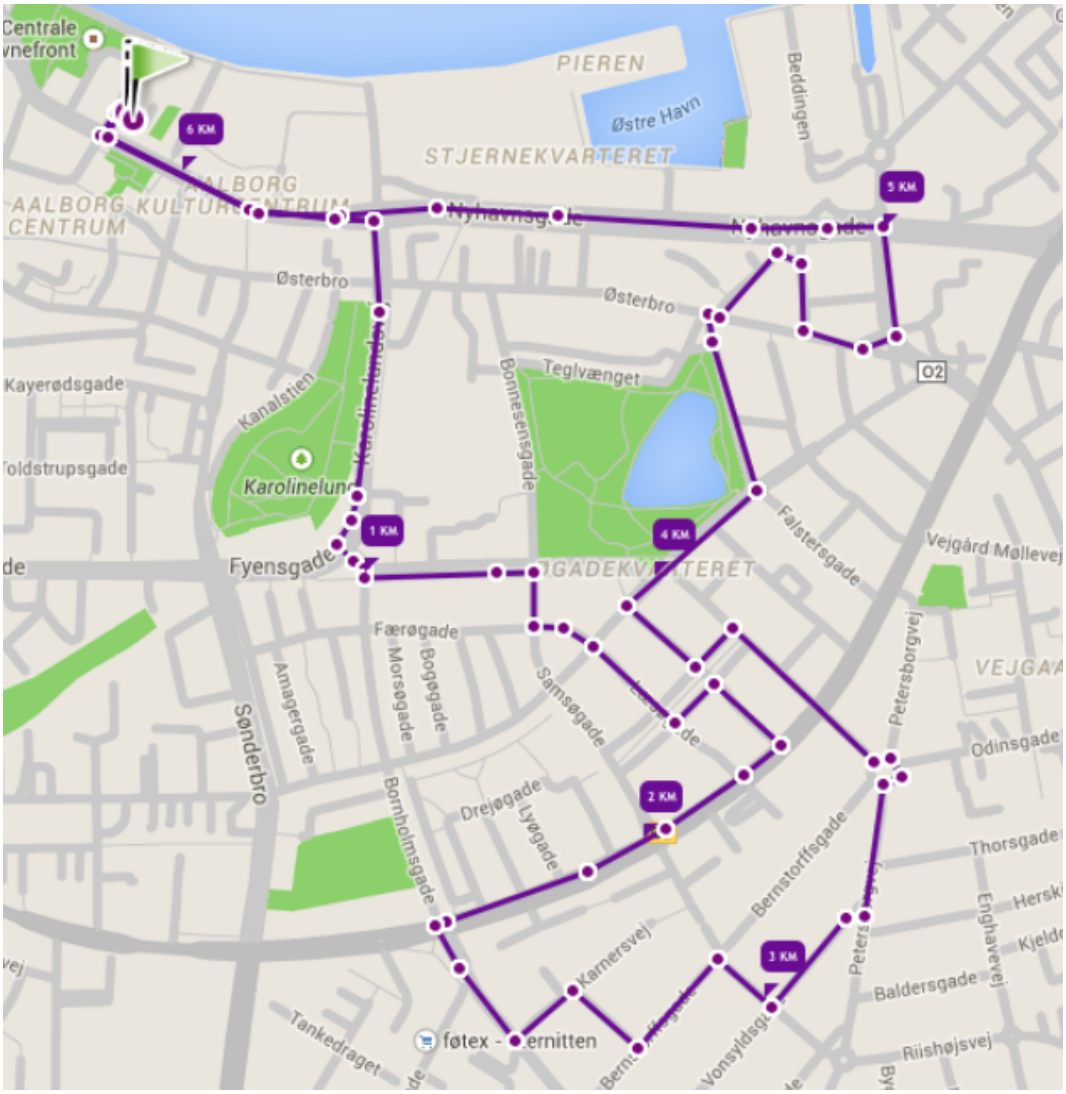


Figure A.1: The route driven by participants in the field research experiment.

Appendix B

City Car Driving Simulator Routes

Figure B.1 below shows the different routes driven by the participants in the interface experiment, as presented in Chapter 9. The green route represents the modality reaction time experiment in Section 9.5. The duration with/without feedback experiment in Section 9.6 is the red route. Finally the counting signs experiment in Section 9.7 is the yellow route.

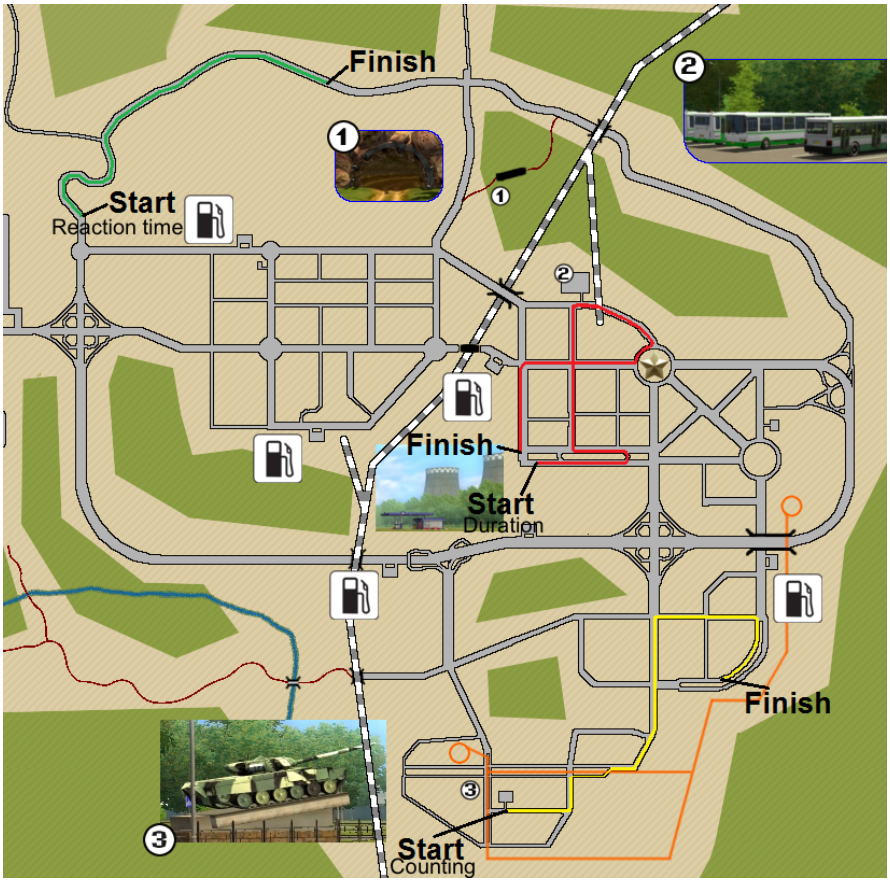


Figure B.1: The route driven by participants in the field research experiment.