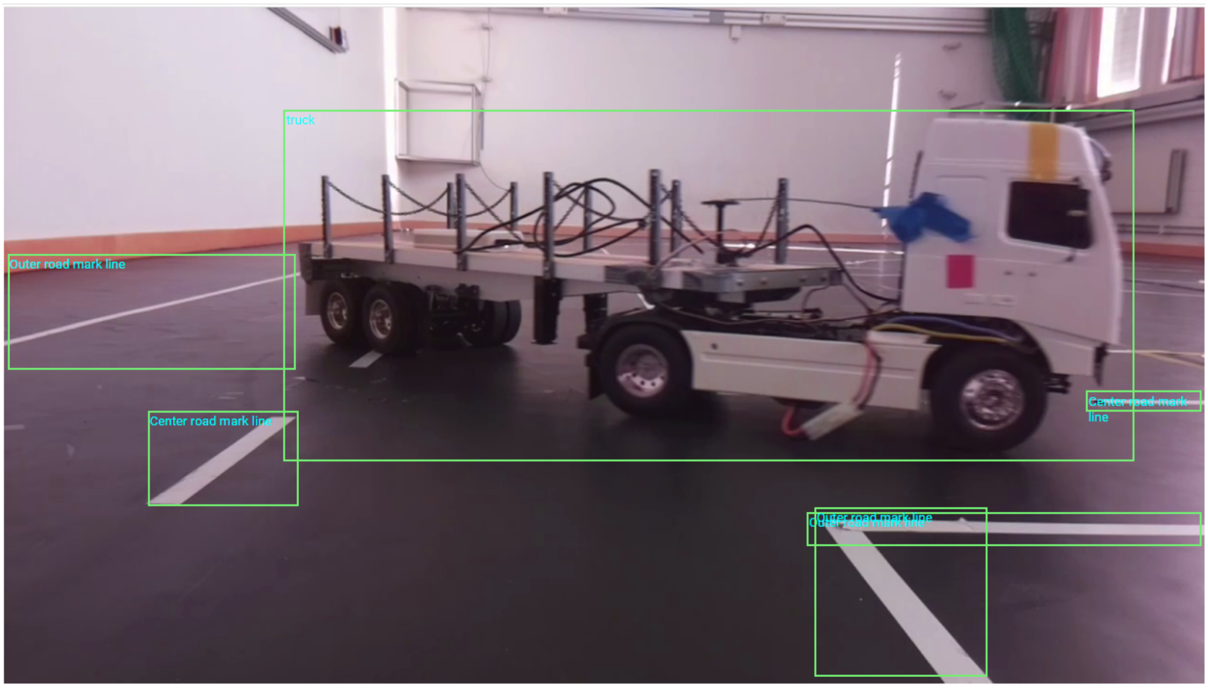


Geo-temporal Online Analysis of Traffic Rule Violations

Bachelor's thesis in Computer Engineering and Information Technology

Davidsson Adam, Fatih Dyako, Larsson Simon,
Naarttijärvi Jesper, Nilsson Daniel, Svensson Marcus

Department of Computer Engineering
and Information Technology

CHALMERS UNIVERSITY OF TECHNOLOGY, UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2020

BACHELOR THESIS DATX02-20-39

Geo-temporal Online Analysis of Traffic Rule Violations

Davidsson Adam, Fatih Dyako, Larsson Simon,
Naarttijärvi Jesper, Nilsson Daniel, Svensson Marcus

Department of Computer Engineering and Information Technology
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2020

Geo-temporal Online Analysis of Traffic Rule Violations

Davidsson Adam, Fatih Dyako, Larsson Simon,
Naarttijärvi Jesper, Nilsson Daniel, Svensson Marcus

Supervisor: Elad Michael Schiller, Department of Computer Engineering and Information Technology

Examinor: Marina Papatriantafilou, Department of Computer Science and Engineering

Bachelor thesis DATX02-20-39
Department of Computer Engineering and Information Technology
Chalmers University of Technology
University of Gothenburg
SE-412 96 Göteborg
Sweden
Telephone +46 31 772 1000

Cover: Image of truck and track used for training a neural network.

Abstract

Due to inattention and not complying with traffic regulations, human error accounts for roughly 94% of all traffic accidents. To counter this, the need to develop systems that can identify traffic rule violations and calculate the risk of collisions. The information reported can then be used to implement preventive measures. Modern vehicles are equipped with sensors and cameras thus making this possible, but it comes with the complication of not violating the privacy of individuals when gathering information.

This project presents a prototype system comprised of three subsystems with the intention of reducing traffic accidents. The first two revolve around the detection of traffic violations with the use of real-time object detection and intention aware risk estimation. The purpose of the third subsystem is to detach personal information from the data gathered by the previously mentioned subsystems. This makes it possible to use the data to pinpoint problematic areas in a traffic environment.

Evaluation of the system was performed in both a simulation environment and with analysis of video feeds from a lab environment. The results of the evaluation show that the prototype system developed in the project is sufficiently accurate to be further developed and implemented for use in real vehicles.

Keywords: Traffic rule violations, Risk estimation, Privacy preservation, Computer vision, Deep learning neural net

Sammandrag

Mänskliga misstag står för ungefär 94% av alla trafikolyckor. Detta på grund av både ouppmärksamhet och brott mot trafikregler. För att motverka detta problem finns det ett behov att utveckla system som kan identifiera trafikbrott och riskabla situationer och rapportera in detta till myndigheter. Den rapporterade informationen kan sedan användas i förebyggande arbete mot trafikolyckor. Moderna fordon är utrustade med sensorer och kameror som möjligör insamling av sådan information, men insamlingen riskerar att äventyra individers integritet.

Detta projekt har utvecklat ett prototypsystem med avsikten att minska mängden trafikolyckor. Systemet består av tre delsystem. De två första syftar till att upptäcka trafikbrott och riskabla situationer med hjälp av realtids objektdetektering och avsiktsmedveten riskestimering. Syftet med det tredje delsystemet är att ta bort personlig information från den data som samlats in av de tidigare nämnda delsystemen. Detta gör det möjligt att använda informationen för att kartlägga problematiska områden i trafiken.

Utvärdering av systemet utfördes i en simuleringsmiljö och med analys av videoflöden från en labbmiljö. Resultaten av utvärderingen visar att prototypsystemet är tillräckligt effektivt för att motivera vidareutveckling och senare implementering i riktiga fordon.

Nyckelord: Trafikbrott, Riskestimering, Integritetsskydd, Datorseende, Djupinlärningsnevronnät

Acknowledgements

We would like to acknowledge everyone who has helped us during this project. We want to thank our supervisor Elad Michael Schiller who, with his knowledge and drive, has helped us throughout the project with great advice and leadership.

We extend our gratitude to Burkin Günke, who helped us with the lab equipment during the entirety of the project. Also, a thank you to Daniel Karlberg who assisted us with his expertise in the subject during the early phases of the project.

Nomenclature

AMT (Automatic Marking Tool) - *Automated tool that uses the trained Neural Net to mark up further training and validation data that can be manually reviewed and then used for training.*
ANN (Artificial Neural Network) - *Series of algorithms used to recognize patterns in sets of data. The process mimics that of the human brain.*
CNN (Convolution Neural Net) - *Specific type of Neural Net used in Deep Learning with supervised training, mainly in processing visual imagery data.*
DL (Deep Learning) - *Subset of machine learning utilizing Neural Networks on large diverse data sets to solve complex problems.*
CPU (Central Processing Unit)
GPU (Graphics Processing Unit)
GPS (Global Positioning System) - *A satellite-based positioning system*
IARE (Intention-Aware Risk Estimation) - *An algorithm for calculating risk in crossings*
MOT (Marking Object Tool) - *Cloud hosted web-app used to mark bounding boxes on test and validation data to be used in Neural Net training.*
MSSM (Markov State Space Model) - *A state based model that uses dependencies between states to predict future states.*
NN (Neural Network) - *Used interchangeably with ANN. Referring to an Artificial Neural Network, unless clearly specified not.*
Randomized response - *A survey technique for eliminating evasive answer bias*
ROS (Robot Operating System) - *A middleware for robot software development*
RTT (Remote Training Tool) - *Web interface for running Neural Net training on a remote server.*
R-CNN (Regions with CNN features) - *CNN strategy for firstly identifying areas of interest by selective search and thereafter processing these areas.*
TVD (Traffic Violation Detection) - *Algorithm for detecting traffic violation scenarios.*
YOLO (You Only Look Once) - *Object Detection Algorithm*

Contents

List of Figures

List of Tables

1 Introduction

Vehicle safety has been increasing steadily over the last 50 years [1], to the point where driver error is the cause for 94% of all traffic accidents [2]. The overwhelming majority of traffic accidents, close to 75%, fall into the two categories driver inattention and driver decisions not complying with traffic rules and/or conditions [2]. To decrease the risk of human errors' impact on traffic safety these two issues stand out as prime candidates to address. Automated systems aiding the driver could prove crucial in improving traffic safety. First off, it could be beneficial to develop systems that identify traffic rule violations by drivers to combat decision errors. This, in tandem with systems that calculate the risk of collision with nearby vehicles and the surrounding environment, to assist in cases of inattention. These solutions could be implemented at no increased production cost, as new vehicles come equipped with the required sensor capabilities from the factory.

In recent years great advances have been made in vehicle automation and automated safety systems [3]. A product of this has been a rapid increase in proximity and camera sensor density of newly produced motor vehicles. With this comes a greater ability for data collection of factors that may affect traffic safety [4]. This data can be used to identify and track the frequency and severity of traffic rule violations in a particular area. Thereby indicating if further actions or resources may be beneficial in order to improve traffic safety.

Modern vehicles collect a vast amount of information about the driver and the surrounding environment [4]. With all information collected, it is important not to violate the privacy of individuals in such a way that they can be individualized. Information needs to be sanitized by the system so that it does not conflict with EU:s (European Union) law about GDPR (General Data Protection Regulation) [5]. GDPR restricts the storage of information about an individual (*e.g.,* an individual's name, location, ethnic origin, and medical condition) should be unknown when a report is filed. A violation of GDPR can lead to fines up to 20 million Euro or 4% of the annual turnover [6].

1.1 Purpose and task description

The purpose of this project is to create technology for increased traffic safety and thereby decrease the number and severity of road-related accidents. To accomplish this, our goal is to implement a system prototype for detecting traffic rule violations and related road risks. Without exposing identities of the individuals involved, the system analyzes the data and compiles it into a report which then can be used to identify critical sections in a road network.

Three different tasks to achieve the described system have been identified as:

- **Detection of traffic rule violations using computer vision** This task consists of developing and evaluate a system that can via computer vision technology identify objects (*e.g.,* vehicles and infrastructure) and scenario markers (*e.g.,* road markings). This system runs locally utilizing vehicle sensor array to produce real-time object velocity and geo-temporal estimation data as well as the presence of a scenario marker. This combined data is used to determine the occurrence of traffic rule violations in close proximity to the test vehicle. The data used to develop this system will be gathered using model trucks in a lab experiment set-up. Initially we limit traffic rule violations scenarios to vehicles crossing a solid line. This will be expanded upon should project timeline allow.

- **Detection of risky traffic situations in vehicular systems**

  Recognizing risks in traffic could be crucial to prevent an actual accident from happening. Therefore a state of the art risk estimation technology [7] will be implemented so that we can detect when a car approaches a dangerous traffic situation. The risk will be determined by estimating the probability that an opposing car's intended maneuver will decline from its expected maneuver. This prototype system will be developed and tested in a simulation environment where the risk is known. Furthermore the prototype system will focus on detecting risky situations in a simulated intersection.

- **Privacy preserving online reporting of geo-temporal information** Mapping incidents of traffic rule violations as well as high-risk situations. This should be done with personal confidentiality in mind so that no private information is leaked during the process. The outcome of this challenge aids the advancement of cost-effective casualty-prevention policies and traffic engineering strategies.

1.2 Our contribution

This project created a prototype system which identifies some traffic violations and traffic risks in some traffic situations. The system then compiles into statistical data that can help with improving road safety, all without compromising personal privacy. This was achieved using state of the art algorithms and can potentially be implemented on modern vehicles using on-board technology. This is the first time that a system allows both the detection of traffic violations and traffic risks, which also offers a safe and GDPR compliant reporting method. The techniques and testing methods used in the development of this prototype system can be used for further development and research on risk and violation reporting systems.

Using machine-learning techniques such as YOLO the system is able to detect traffic violations in some traffic situations using image streams from existing car cameras. Furthermore, the system can determine dangerous traffic situations in intersections using Intention-aware risk estimation. The system can report traffic violations and dangerous traffic situations using a software-based randomized response implementation, without violating personal privacy.

Tests were performed on all subsystems. The tests were conducted with the following tools and methods: The traffic rule violation detection system was tested using the lab environment equipment and computer vision with YOLO; The risk detection subsystem and the project system was tested using ROS and Gazebo; The privacy preserving reporting subsystem was tested using simulated data. Testing showed that the system achieved the capabilities described in the task description.

Traffic Rule violation detection was done with computing current positions and previous positions of vehicles in correlation to the surrounding environment, such as solid lines. The Risk detection subsystem successfully detected all risky situations in the 96 test cases performed. This was done with a margin that makes it possible to use automatic braking systems to mitigate traffic accidents. Furthermore, the testing of the randomized response reporting showed that it was successfully implemented.

With these results the project concluded that these technologies has potential to be useful in future systems.

The code and documentation can be found on the projects' GitHub repository following this URL `https://github.com/JesperNaarttijarvi/DATX02-Vehicle-Automation`.

2 Theory

The purpose of this chapter is to present and explain the theory that the project is built upon. To get a better understanding of the system architecture (chapter 4), this section gives a rundown of methods and techniques that are used to implement said system architecture.

2.1 Image processing techniques

In order to detect traffic rule violations, we utilize Computer Vision [8] working on the data stream coming from Visible Light Cameras on vehicles. This allows us to identify objects, of our own choosing, trained in our Deep Learning Artificial Neural Network (DL-ANN). This Artificial Neural Network (ANN/NN) [9] is used both for object detection and object tracking. Determining if objects that may be used to propose traffic situations are present and their spatial relation to each other. In order to process video in real-time with limited compute capability using the right algorithm as well as the purposeful strategy in objects to detect had to be considered. As this is paramount to achieve in high-speed vehicle traffic.

An Artificial Neural Network combined with an algorithm analyzes the image frames of the video stream represented as matrices of RGB color values. Throughout stages convolutions are applied on the neighboring pixels, in most cases by a 3x3 grid, around said image pixel. The product of these equations is used to identify edges, shapes, and eventually higher-order representations. It outputs what objects are present in the image and with what certainty as well as their position in the image.

Depending on what algorithm is used this is done differently and there have been many improvements in this area starting in 2012 when Computer Vision transitioned over to Deep Learning. Earlier strategies such as Convolution Neural Net (CNN) [10] apply convolutions with full coverage of pixels in multiple steps for all nodes in the NN. Later advancements in Regions with CNN features (R-CNN) [11] utilize randomly positioned fields and try to detect similar color, texture or pattern within these, highly probable to compose an object. After this, it only performs the taxing CNN in these areas. Leading up to the current state-of-the-art strategy You Only Look Once (YOLO) [12] which also first produces a class probability map and focuses on those areas with multiple cycling of convolutions and pooling. YOLO is also integrated in an open source neural network framework, Darknet [13]. The performance of YOLO in conjunction with this ready to use framework with ample documentation makes it the prime candidate for use by our Image Processing subsystem.

These Neural Networks have to be trained with training data in order to later be utilized for detection. Training data is produced by taking large sets of image data of the classes and manually defining the region where they are present. This combined data of image and region therein a named class is located is fed to the Neural Network for training and improves the weights used for better accuracy in detection. Weights are adjusted so that the ones in accordance with proof positive are rewarded and ones in opposing are punished. Incrementally strengthening the weights ability to correctly detect the trained classes even if in high variance from the training data.

YOLO (You Only Look Once) is based on a single neural network constructed from the ground up for object detection. This enabled high-performance detection which in turn makes it highly suitable for real-time video processing. Another benefit of YOLO is that it learns very generalized representations of objects and can, therefore, identify objects with some certainty even if conditions vary greatly from the conditions it was trained under. The object classifiers of YOLO also work in a more global and contextual way than previous techniques making it much more adept in understanding what part of a marked object in training is the part of interest. Making it much less likely to produce false positives by not misjudging 'dead space' or background as part of the classifier.

YOLO processes images in a single pass, hence the name, You Only Look Once. The first step in this process is dividing the image up into a S x S grid cells, in YOLO9000 this was changed to 13 x 13 as it proved to improve detection of small objects over fewer cells. Within these cells, multiple bounding boxes of possible objects and confidence scores that these match classes trained to be detected by the model are produced. The confidence score is a combination of how probable the model predicts that an object class is present within and how accurately the bounding box is to contain the entire object.

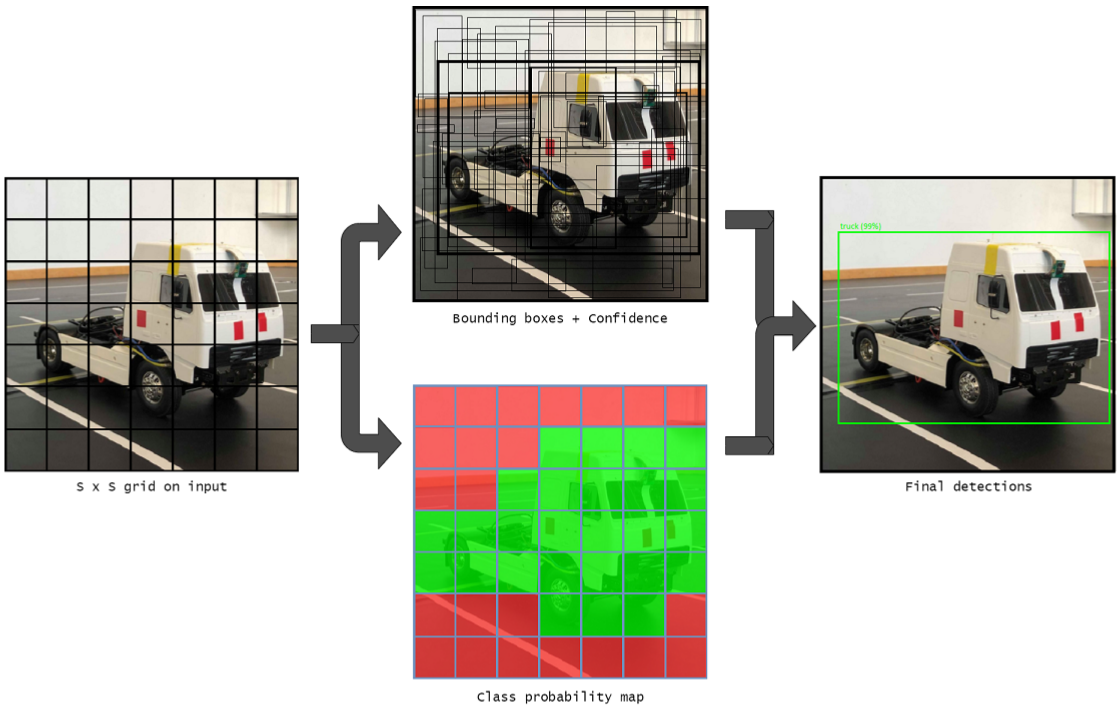


Figure 1: YOLO Process

YOLO [14] was first introduced in 2016. It was later iterated upon that same year with YOLOv2 [15] also known as "YOLO9000: Better, Stronger, Faster" when it was combined with a purpose-built CNN framework named Darknet [13]. In 2018 the current version YOLOv3 [12] was released with improvements in both accuracy and performance.

2.2 Intention-aware risk estimation

The Intention-Aware Risk Estimation (IARE) is used in the Risk detection subsystem to reason about traffic situations and collision risk [7]. In this section, the theory behind Intention-Aware Risk Estimation for General Traffic Situations will be explained.

There are many approaches on how to tackle the problem with risk estimation [7]. The conventional solution relies on trajectory prediction, meaning that a trajectory is assessed based on the speed and position of vehicles [16] [17]. The predicted trajectory is then compared between all vehicles to detect potential collisions. This method is very resource intensive which prevents real time safety systems. A less resource intensive alternative is Intention-Aware Risk Estimation. IARE which is the approach taken in this solution, reasons at a semantic level [7]. Risk calculation is based on the intentions and expectations of drivers. More precisely, high-risk traffic situations are determined by comparing the intentions of drivers with what they are expected to do.

The IARE algorithm has been shown to be capable of detecting dangerous situations early [7]. For instance in simulations of 240 dangerous traffic situations every collision predicted at least 0.6 s before collision and in 80 of the cases the collision was predicted 2 s before collision. Furthermore the IARE algorithm has been shown to comply with real-time constraints. In field trials the 20 non-dangerous tests caused no false alarms and in all of the 90 dangerous tests the system warned the driver early enough so that the collision could be avoided by braking. Therefore, the IARE algorithm is suitable to use in the Risk detection subsystem.

2.2.1 Markov State Space Model

As mentioned before, the approach is to estimate the intention of drivers and then compare that to what is expected of them. In the IARE algorithm, a Markov State Space Model (MSSM) is used to predict the traffic situation semantically [7]. A MSSM is a state-based model that uses dependencies between states to predict future states. The MSSM used can be seen in Fig 2 below.

The first layer corresponds to the expected behavior of the vehicle. It is implicitly encoded in the multi-vehicle dependencies. For instance, the model assumes that a vehicle will stop if it approaches an intersection where another vehicle has right-of-way. This expectation is represented by the variable E_t^n. E_t^n is a conjunction of variables. Furthermore, E_t^n is derived from the previous situational context. Each variable in the conjunction I_t^n has a corresponding variable in E_t^n.

The second layer corresponds to the maneuver performed by the vehicle. The variable I_t^n then represents vehicle n's maneuver at time t. This layer is called the "Intention" of the driver and it is dependant on the variable E_t^n, as mentioned before, since the model assumes vehicles will follow the traffic rules with a high probability.

The third layer corresponds to the physical state of the vehicle. For instance position, speed, etc. The conjunction of the variables at this layer is represented by ϕ_t^n. Which therefore represents the physical space.

The lowest layer corresponds to the available measurement. These measurements consist

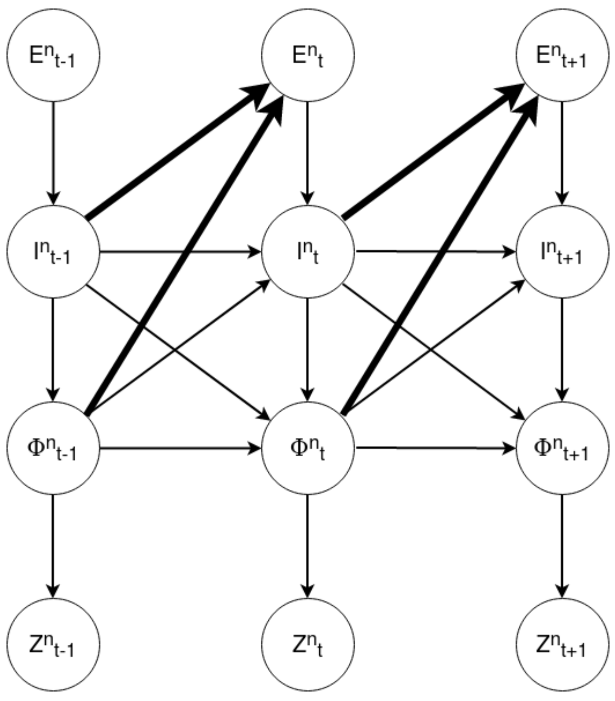


Figure 2: Markov State Space Model, bold arrows represent multi-vehicle dependencies.

of position, orientation and speed of each vehicle and are represented by the variable Z_t^n.

2.2.2 Intended maneuver

The intended maneuver tries to assess how a driver will operate a vehicle through a crossing. As mentioned in the report [7], the operation is divided into two separate parts: a lateral and longitudinal motion. The lateral motion represents the course of the vehicle. More precisely, if the driver intends to drive straight through the crossing, turn right or turn left. In this work, however, the focus lies in the longitudinal motion of a vehicle. The longitudinal motion represents the drivers' intention of stopping at a crossing. This intention can be one of the following: *go* or *stop*. If a driver intends to stop at the intersection it will be represented by I_t^n= *stop*. If I_t^n instead is set to *go* it shows that the driver intends to drive through the intersection without the intention of stopping. The intention is continuously calculated using the current expectation Es_t^n and the previous intention I_{t-1}^n of the driver:

$$P(Is_t^n | Is_{t-1}^n Es_t^n) \tag{1}$$

The way I_t^n is calculated, as mentioned in the previous report [7], is as follows:

The assumption made is that drivers have a high probability of following what is expected of them and therefore P_{comply} is set to $P_{comply} = 0.9$

Is_{t-1}^n	Es_t^n	$P(Is_t^n = go)$
go	*go*	P_{comply}
go	*stop*	*0.5*
stop	*go*	*0.5*
stop	*stop*	$1.0 - P_{comply}$

Table 1: Calculation of the intention of a driver

2.2.3 Expected maneuver

The expected maneuver models surrounding vehicle's influence on the maneuver a vehicle performs[7]. The expected maneuver can be divided into two parts: lateral and longitudinal motion. However, the lateral motion is not necessary to model in an intersection. Since the direction a vehicle will take in an intersection will not be affected by other vehicles. The longitudinal motion for vehicle n at time t can defined as: $Es_t^n \in \{go, stop\}$. Where *stop* means that the vehicle will adapt it's speed so it can stop at the intersection and *go* mean that this is not the case. The expected longitudinal maneuver is derived from the previous intended course, pose and speed of all the vehicles in the traffic environment:

$$P(Es_t^n | I_{t-1}\phi_{t-1}) \tag{2}$$

The expected behavior of a vehicle is defined both by law and driver judgment. Therefore this model is based on expected driver behavior. For instance, the necessity for a vehicle to stop in an give-away intersection is calculated a bit simplified in the following way:

1. Project forward the position of the vehicle n until it time t^n where it reaches the intersection using the previous state and a constant speed.
2. Project in the same way for all the vehicles V with right-of-way.
3. find the vehicle $k \in V$ which is most likely to cause the vehicle to stop, by finding the smallest time gap for the vehicle n execute it's behaviour without collision
4. The necessity for vehicle n to stop at the intersection is calculated as the probability that the time gap is not sufficient, using a probabilistic gap acceptance model.

This context-aware reasoning can be applied to other traffic situations and allows us to detect unexpected i.e. risky behaviour.

2.3 Privacy preservation

When collecting and analyzing the data from the risk detection and image processing subsystems, it is critical not to reveal any information about the individual(s) involved in *e.g.* a traffic violation. If the collected data is to be reported to a federal organization, it first needs to be processed. This section presents a technique using randomized response [18] to achieve a privacy preserving reporting system. Randomized response is an approach that can be used in surveys to avoid evasive answer bias. Warner [18] argues that with this method, evasion bias is greatly reduced to the point that it is almost completely removed.

An interviewee is prompted to only answer truthfully on a yes or no question based on a probability. For example the interviewee could toss a coin (see Figure 3). If the coin lands on "heads", then the interviewee must answer the truth. But if the coin lands on "tails", then the interviewee will toss the coin again. Supposing that the second coin toss lands on "heads" the interviewee will answer yes regardless of it being the truth and on "tails" the interviewee answers no. This eliminates the possibility for even the interviewer to know if the interviewee tells the truth. Due to the interviewee's answer only having a probability of being true or false, the interviewee can not be held accountable to the result. Therefore any information gathered by using this method can be used without the concern of jeopardizing individual's privacy.

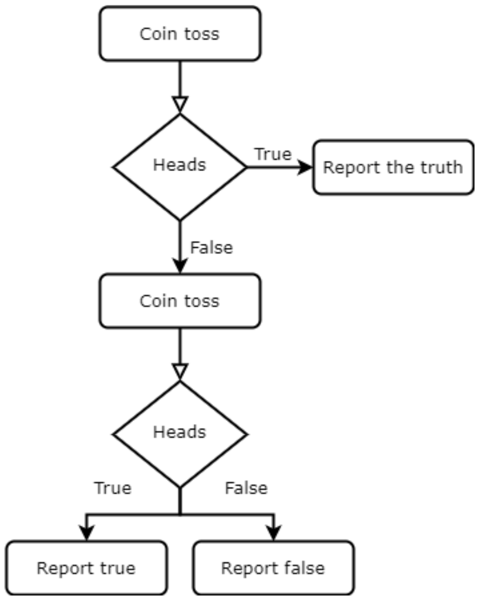


Figure 3: Algorithm flowchart

The use of the randomizing response method was conceived as a way to reduce evasive answer bias for primarily surveys. But its effectiveness of giving accurate statistics and simultaneously being able to detach the interviewee from their answers makes is suitable in aiding the system to comply with the GDPR [6]. The privacy preserving reporting system is automatic, requiring the method being able to be automated in software. The randomizing response method is an excellent choice due to its systematic approach and its capabilities to be adapted for different use cases.

An alternative to randomized response is *k-anonymity* [19]. This method is based on removing attributes from a set of data, *e.g.* an individuals' name, sex, or age. In the case of an individuals' age, the attribute can be replaced with a range. If the individuals' age is 42, this can be replaced with *e.g.* $40 - 45$. However, there is a possibility that individuals' information is revealed if an attacker has prior knowledge of the set of data. This is not an issue with randomized response, since it only returns statistical information completely void of any individuals' information.

3 Software Engineering Process

This chapter of the report intends to elucidate the work methodology of the project, as well as describe the different tools and software that were used throughout.

3.1 Software development methodology

The project group was divided into three separate teams of two people each. Each one was assigned a subsystem to work on. The teams have worked incrementally on the subsystems. The incremental process involved four steps which were cycled. In the first step, the specifications for the current stage of the subsystem were established. For instance, the project requirements were discussed and concertized in a meeting. The next step was to design functions that meet the specifications set by the first step. This was primarily done within the subgroups. The third step was to code the designed functions, which were done in pairs or by one group member. The last step consisted of testing the system to ensure that it satisfied all requirements. This process was then repeated several times to acquire the finished product.

The work process also involved two weekly meetings. The meetings discussed objectives, work progress, problems, design, and specifications. Each meeting was led by a different chair who led the meeting and wrote a protocol. Furthermore, the group also held meetings with the supervisor. During which the group got feedback.

3.2 Workflow and version control

During the entirety of the project, the software produced was continuously integrated with the version control system Git [20]. The three subsystems were developed in one unified repository under different branches. When working on new functions in said subsystems, a new branch was created. After testing to assure the code was bug-free, the branch for this particular function was then merged into the subsystem branch. After the project was finished, the code was uploaded to BitBucket to be used by next year's bachelor project groups [21].

3.3 Software tools and libraries

To realize the project, a diverse set of software tools and libraries were used. The purpose of this section is to present them and explain why they were suitable for this project.

3.3.1 NodeJS

NodeJS is an asynchronous event-driven JavaScript engine, designed for scalable network applications [22]. In this project, a small NodeJS application is used to host tools on a public website to enable remote work with AI and synchronizes the projects' work. NodeJS was chosen for its simplicity and scalability.

3.3.2 VueJS

VueJS is a framework for Javascript and HTML, used to create powerful websites that are scalable and easy to work with [23]. The project used VueJS to create online tools, such as Marking Object Tool for image processing (See chapter 4.3.1). VueJS was chosen because project members previous knowledge with the framework and VueJS's easiness to create and iterate websites.

3.3.3 Python

The majority of the project is programmed using Python for three different reasons. First off, most group members were to a certain degree already familiar with the language. The second reason was that the previous bachelor projects that our project is built upon also used Python as their primary language. Lastly, Python is highly compatible with ROS which makes it easy to evaluate the code written in Python.

3.3.4 Firebase Hosting

Firebase Hosting is a web content hosting service written in NodeJS where a user can quickly deploy web apps/websites [24]. Firebase Hosting was chosen because of the member's knowledge of the software and that it is free to use until a quota is met.

3.3.5 ROS

ROS is a middleware for robot software development [25]. One service ROS provides is message-passing between processes. Separate processes are used in this project to execute different subsystems. Therefore message-passing is used to connect the different subsystems. The message-passing between the processes is presented in a graph architecture where nodes may receive and post data. For instance, the Risk estimator needs the positions and velocity of the vehicles to update the risk estimation which it receives from other subsystems. ROS message-passing creates a way for subsystems to communicate with each other.

4 System Architecture

This chapter begins by describing the system architecture as a whole, followed by a short reflection on a central part of the system and ethics. After that a more detailed description of the three different subsystems and their implementation.

4.1 System overview

The developed system is a traffic hazard reporting system. This system detects dangerous traffic situations as well as traffic violations and reports these to a third party without jeopardizing personal integrity. To achieve this functionality three independent subsystems have been developed and interlinked to form the complete system. The subsystems have the following functionalities:

1. Detecting traffic rule violations. This subsystem is responsible for detecting when a driver doesn't follow the traffic regulations, using computer vision.
2. Detecting dangerous traffic situations. This subsystem is responsible for detecting when a driver makes a dangerous maneuver in traffic, using vehicle position.
3. Privacy reporting of geo-temporal information. This subsystem collects and reports driver's geographical data without incriminating personal integrity.

The systems function could be fully realized by connecting it to a compatible vehicle and an external database. This is however outside the scope of this project's system and will only be explained briefly in this section.

4.1.1 Data flow

The subsystems, the sensors, and the system outputs communicate through ROS. Each subsystem uses a ROS node to publish data for other systems to read. First, the information gathered and then posted by the sensors are read by the system. For instance, the risk detection subsystem reads the input from the positioning system and the traffic violation detection subsystem reads the input of the camera. The risk detection subsystem and the traffic violation detection system then operates on their respective input data. These two subsystems then post their output data and the privacy reporting subsystem reads it. Furthermore, the risk detection subsystems output can also be read by an external automatic braking system which then can act in case of a risky traffic situation. Finally, the privacy preserving reporting subsystem can operate its data and then post to a database or another party.

The system uses two sensors a camera and a positioning system. However, the system is not dependent on a specific type of sensor but could be implemented with any sensors that provide correct information. The camera is used to record an image stream for the traffic violation detection subsystem. As an example, the camera sensor could be a camera used in a self-driving car. The positioning system provides the risk detection subsystem with positional data on surrounding cars. For instance, a sensors detecting other vehicles used for self-driving and a GPS could be used.

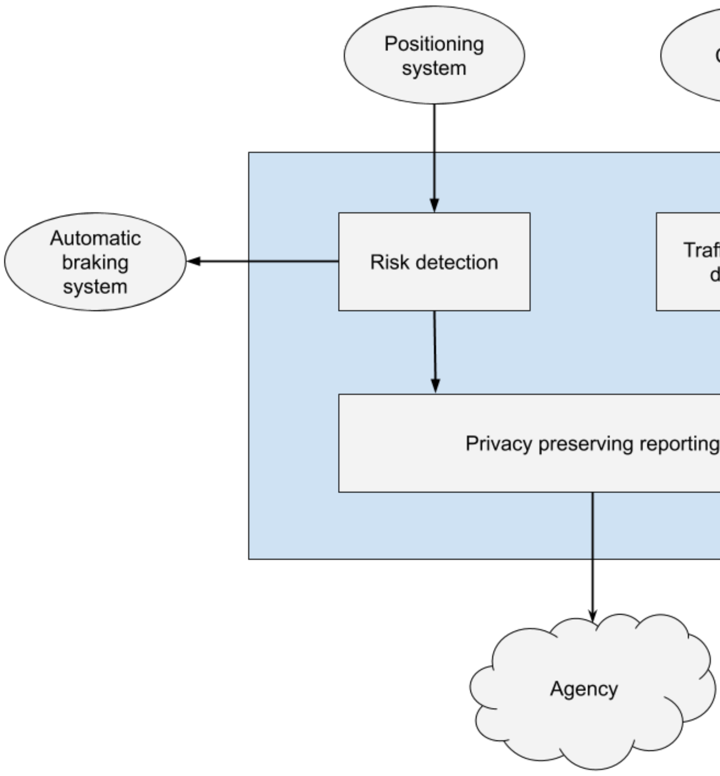


Figure 4: System overview of the complete system

As seen in Figure 4. when data has been processed the system gives two outputs. The first output comes directly from the Risk Estimator and can be used for an automatic braking system. E.g. when the Risk detection subsystem detects a dangerous traffic situation it can not only be used to report the risk but also to activate an automatic braking system to stop the car and prevent an accident. The second output, as seen in Figure 4, goes through the Privacy reporting subsystem and is connected to an agency. This means that the collected data can be sent to, for example, Trafikverket. They can then analyze this data to detect hazardous sections in traffic and modify the road network to eliminate these sections.

4.1.2 Ethical system architecture

The structure of the system architecture in this project was largely influenced by ethical considerations. These ethical considerations concerned both traffic safety and privacy. In this section, we will explain the ethical considerations of the system architecture.

As mentioned in the background human error is the cause of 94% of all traffic accidents [2]. However, these accidents could also be viewed as a systemic problem instead of being a problem of individual bad drivers. In some traffic environments, such as a country road, it could be challenging to drive safely. The problem then is not the individual driver but the system of roads. Therefore, the system architecture of this project works to create a safer traffic environment by improving the traffic system. This is done both directly in risky traffic situations and preventative. The risk detection subsystem could trigger an automatic braking system during risky situations. The reporting of risky traffic situations and traffic violations by the system could aid the government agencies to improve traffic environments. However, reporting to external parties creates ethical problems concerning privacy. For instance, private information could be abused by a malicious actor. Therefore the system architecture also contains a privacy preserving subsystem. This subsystem gives the system the ability to report to other parties without compromising privacy.

4.2 Risk detection

The Risk detection subsystem calculates the risk of collisions with other vehicles. The Risk detection subsystem communicates with other subsystems with ROS message-passing, it calculates the risk using a previously developed Risk Estimator and feeds the Risk Estimator with correctly formatted data in manageable intervals. The Risk Estimator analyses surrounding cars to preemptively determine their intentions and detect potential collisions, which happens when drivers make unexpected maneuvers. This is done by calculating the expected maneuver for a driver and continuously check that the driver follows this maneuver. This section will go into detail of how this subsystem works and how it has been implemented in our system.

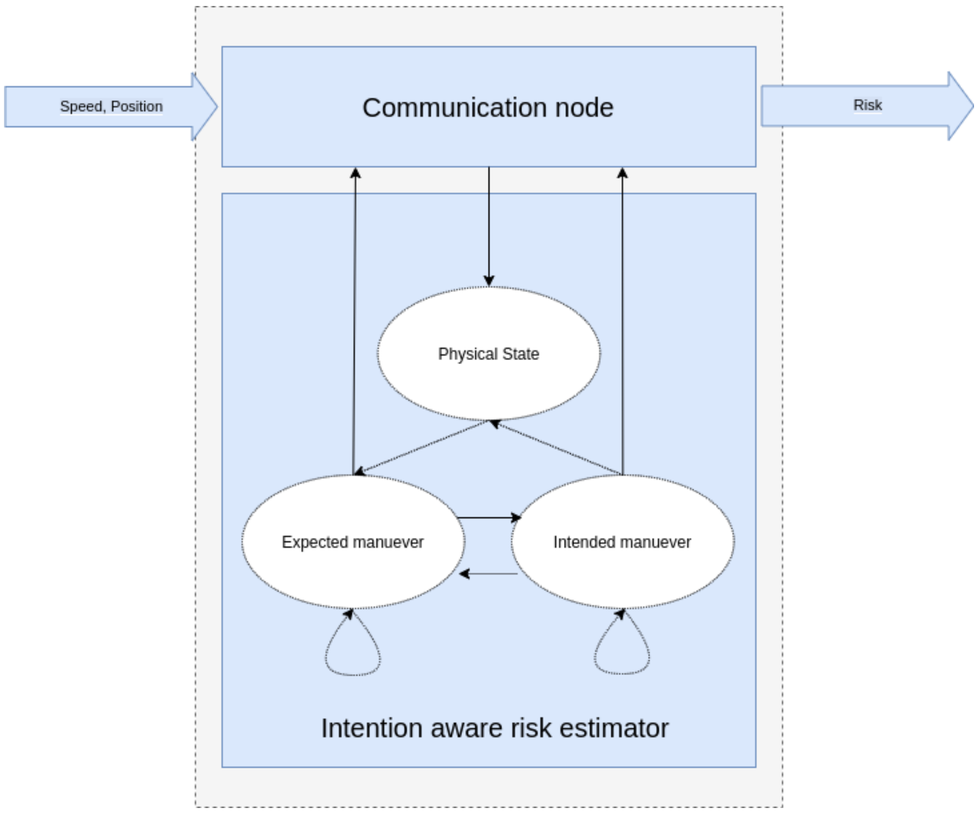


Figure 5: Risk detection subsystem

4.2.1 Physical state and measurements

The Risk estimator tracks the physical state of the model [7]. This physical state is based on the available measurements. The model continuously gains more measurements and to improve on the accuracy of the model. However, in real-world scenarios the sensors measuring also registers noise. Therefore the model uses statistical analysis to predict the best measurement. The physical model is used to calculate the expected maneuver.

4.2.2 Expected maneuver

The algorithm starts by calculating what the cars expected maneuver is, which can either be ‘stop’ or ‘go’[7]. This is done by first checking if there are more than one vehicle approaching the intersection. If there are multiple vehicles approaching, the Risk Estimator uses priority lanes to decide which cars are expected to go and which are expected to stop. The Risk Estimator also tries to assess the time it will take for the cars to reach the intersection, using their speed and position. If the gap between the cars is big enough multiple cars can be expected to go even though they approach from different directions.

4.2.3 Intended maneuver

The algorithm uses the speed and position of the cars to calculate their intended maneuver, which consists of either ‘go’ or ‘stop’ [7]. It also uses the previously intended maneuver and the current expectation since drivers are inclined to follow the law and rather unlikely to drastically change intention. The intention, alongside the expectation, is then later used to assess risk.

4.2.4 Communication node

The communication node is built upon message-passing using ROS. This node collects the physical state of each model in the simulation and reformats the data so that it correlates with what the Risk Estimator is expecting. After the Risk Estimator has evaluated these values a ROS message is sent back via the communication node. This message consists of an integer that represents the severity of a traffic situation. I.e. a high number shows that a driver, with high probability, will deviate from it’s expected course and potentially cause a collision.

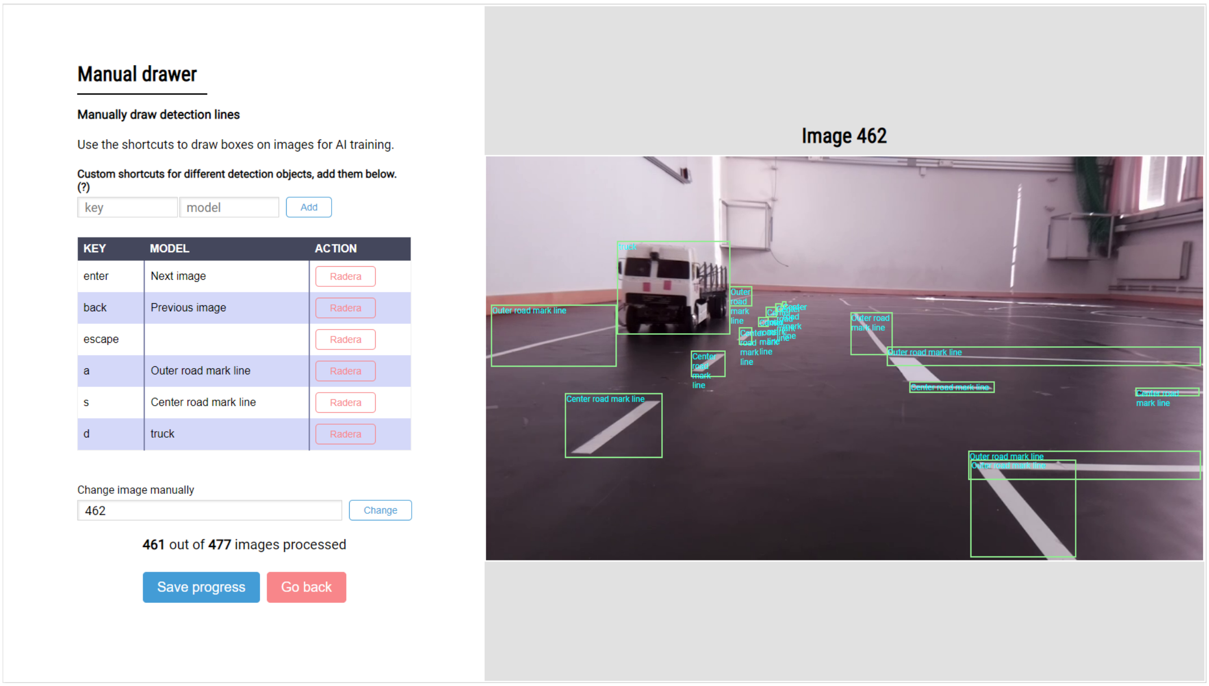
4.3 Image processing

Image processing consists of three subsystems. One system is used to preprocess and convert videos and images for the YOLO Neural Network. One system is used to remotely train and validate the YOLO Neural Network. The last system is fed the processed video-stream of the YOLO Neural Network to determine if a traffic violation has occurred. Training a reliable Object Detection Neural Network requires vast amounts of images names in rectangular boxes (labels) with coordinates surrounding the objects the Neural Network is trying to detect. For each new object the Neural Network is trying to detect, the complexity increases. To train a reliable Neural Network it is required to have at least 2000 images of each object [26]. To get the required amount of images a program was needed to convert videos into images for labeling, hence Marking Objects Tool was created.

4.3.1 Marking Object Tool

The Marking Object Tool (MOT) is a program developed to convert videos into images and to manually mark each image with object names in rectangular boxes (labels) for the YOLO Neural Network to train upon. To enable work from remote locations and also synchronizing the projects progress MOT is hosted on Google's Firebase Hosting (see section 3.3.4) hosting services and the program's frontend is written in VueJS (see section 3.3.2) and backend written in NodeJS (see section 3.3.1).

The program allows a user to create projects with images and videos of the objects the user wants to detect. The system processes and converts the videos into images with a variable delay between each frame and makes the project public, this makes it possible for all users to interact and work with the project. Each image in the project is ready to have its objects labeled manually, see Figure 6 for an example of marking the objects "truck", "inner road marking line" and "out road marking line". After the users have labeled the images in the project it is available for download. When downloading the project the program converts all labels into text files for each corresponding image where each text file contains the coordinates and size of each object on the corresponding image. The downloaded project images and text files are ready to be used to train a YOLO Neural Network.


Manual drawer

Manually draw detection lines

Use the shortcuts to draw boxes on images for AI training.

Custom shortcuts for different detection objects, add them below. (?)

KEY	MODEL	ACTION
enter	Next image	Radera
back	Previous image	Radera
escape		Radera
a	Outer road mark line	Radera
s	Center road mark line	Radera
d	truck	Radera

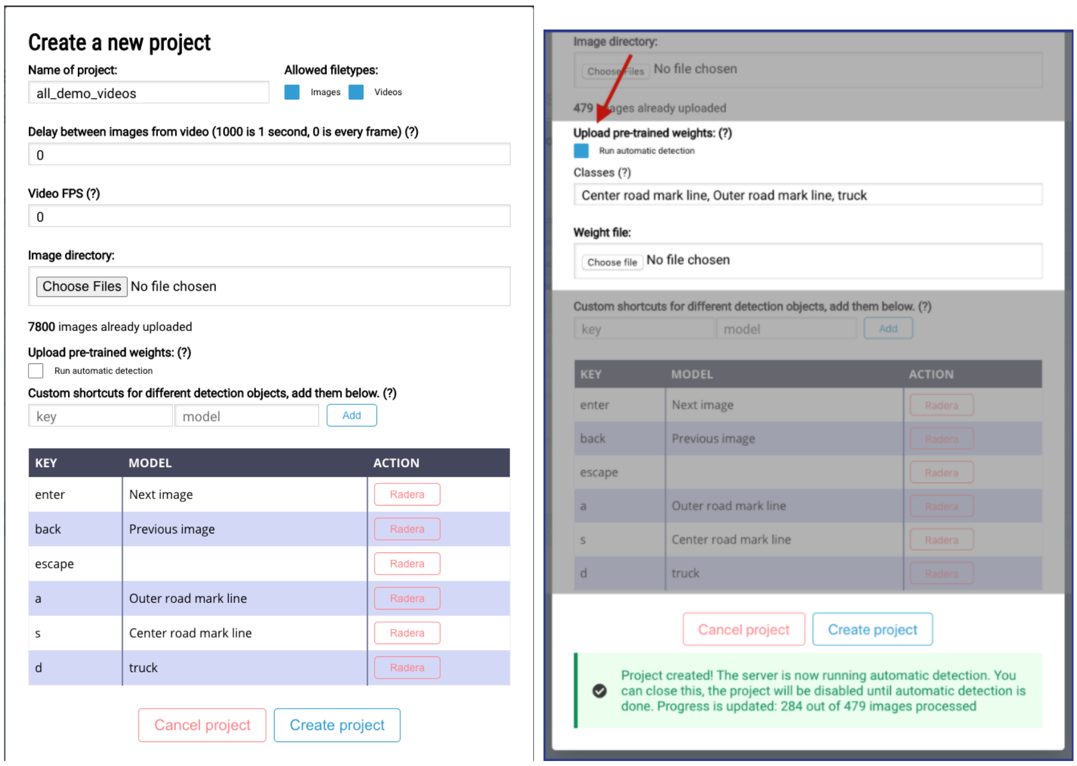
Change image manually

462

461 out of **477** images processed

Figure 6: Marking Object Tool, marking view

Marking Object Tool also contains another subsystem within, called Automatic Marking Tool (AMT). AMT is written in VueJS and NodeJS. The program works similarly to the Marking Object Tool and is hosted online together with the Marking object tool on Google's Firebase Hosting. However, instead of marking labels manually, AMT uses a pre-trained YOLO-weight to create labels automatically with a YOLO Neural Network, the interface can be seen in Figure 7. AMT makes it much faster to increase the accuracy of the YOLO Neural Network and makes it easier to adapt the Neural Network to new environments since the heavy lifting is done by the pre-trained YOLO-weight and YOLO. AMT can be enabled when creating a project and allows the user to upload a pre-trained YOLO-weight. AMT converts the videos into images and applies YOLO Neural Network Object Detection with the given weight to identify and create labels for each detected object in each image. Afterward, the user can iterate through the images and manually change the labels AMT created. After the program has completed its calculations the user can download the project images and labels and feed it directly to train YOLO.



Create a new project

Name of project: all_demo_videos

Allowed filetypes: 🟦 Images 🟦 Videos

Delay between images from video (1000 is 1 second, 0 is every frame) (?): 0

Video FPS (?): 0

Image directory: Choose Files No file chosen

7800 images already uploaded

Upload pre-trained weights: (?)

☐ Run automatic detection

Custom shortcuts for different detection objects, add them below. (?)

key | model | Add

KEY	MODEL	ACTION
enter	Next image	Radera
back	Previous image	Radera
escape		Radera
a	Outer road mark line	Radera
s	Center road mark line	Radera
d	truck	Radera

Cancel project | Create project

(a)

Image directory: Choose Files No file chosen

479 images already uploaded

Upload pre-trained weights: (?)

🟦 Run automatic detection

Classes (?): Center road mark line, Outer road mark line, truck

Weight file: Choose file No file chosen

Custom shortcuts for different detection objects, add them below. (?)

key | model | Add

KEY	MODEL	ACTION
enter	Next image	Radera
back	Previous image	Radera
escape		Radera
a	Outer road mark line	Radera
s	Center road mark line	Radera
d	truck	Radera

Cancel project | Create project

Project created! The server is now running automatic detection. You can close this, the project will be disabled until automatic detection is done. Progress is updated: 284 out of 479 images processed

(b)

Figure 7: Marking Object Tool, automatic marking

4.3.2 Remote Training Tool

Training and validation of a YOLO Neural Network require a powerful computer with a modern expensive CUDA-based graphic card and a powerful multi-threaded CPU. Training and validating require a lot of time to complete, basically locking the computer until it finishes. To combat this and remove the user to have a powerful computer, the remote training was created. Remote Training Tool (RTT) is a program written in NodeJS, VueJS, and Python to enable training to be run on a remote server, controlled with an interface on a website. RTT eliminates the need for a user to lock up their computer for several hours and makes it easier to control the process via a web interface. The web interface also enables the user to work from remote, the user can start a new batch of training and validation of a YOLO Neural Network and then leave it working for hours at a time, without the need to constantly check in on the progress being made. After the training is done the user can download the finished Neural Network.

All these three subsystems are used in symbiosis, to train and create a new Neural Network that's able to detect custom objects, like Road Markings and Trucks in this case. Usually, the user uses MOT to mark a set of hundreds of images manually, then trains a neural network to detect the set of objects. The newly trained neural network is then used with AMT to run on thousands of images to mark them. The user will manually go through all the automatically marked images and correct the few mistakes the AMT made. Lastly, the user will train a new Neural Network with RTT and the thousands of marked images. This process is repeated until a Neural Network with enough accuracy is achieved, increasing the dataset each time.

4.3.3 Traffic Violation Detection

Traffic Violation Detection (TVD) is a program written in Python used to determine if a traffic violation has occurred. The program uses a video-stream from the test vehicles (see section 5.3.2), which in turn is fed to the TVD where it's processed by the YOLO Neural Network and the Neural Network identifies road lanes and vehicles. The identified objects and their current position is used by the TVD to calculate if the vehicle has committed any violations, like driving over a solid white line. To calculate if a vehicle has committed a violation TVD calculates the position of the vehicle in correlation to the road lines and compares it to previous frames. If TVD lost sight of a solid white line and sees that a truck has taken its place, a violation has been detected.

4.4 Privacy preserving reporting

The reporting module receives data from the traffic violation and risk detection subsystems to then compile the data into a coherent report. The data must in no way be traced back to the individual or individuals that were involved in a violation or risky situation. To ensure that the collected data does not violate personal privacy, the system utilizes the randomized response [18] algorithm and omits all visual and sensitive information. Therefore, it complies with GDPR [6] and can be used to locate problematic areas in a road network. This section will go into detail how this subsystem is implemented.

4.4.1 Data parameters

The received data is a bit different depending on which subsystem is using the module. The risk estimator uses coordinates to locate surrounding vehicles and is able to pass on location data. On the other hand, the traffic violation detection system uses image detection and does not inherently collect any location data, therefore the location parameter may be gathered elsewhere. It can be passed from the risk estimator's coordinate system or a GPS system common in most modern vehicles. There are 4 parameters that are required by the reporting module. The first parameter is the violation type *e.g.* give way or the scenario the risk estimator reports on. The second parameter is the location which is based on the location name and/or a coordinate. The third parameter is a timestamp. Forth and final parameter is an occurred marker, a value of '1' means that the violation in the first parameter occurred and the value of ‘0’ means that the violation did not occur. If the data is formatted correctly, then the reporting module will take the string of data, split it into a list of tuples and then passes it to the randomized response [18] implementation. An example string of data that the reporting subsystem would interpret is:

```
violation:give way, location:korsvägen, pos:[330,051],
timestamp:2020-04-10 15.32, occurred: 1
```

No visual data of drivers or vehicles are stored in this system. Therefore, the stored parameters shown above does not violate the GDPR [6], since the information can not be associated with an individual. This is due to the omission of personal information that could link a driver or vehicle to an event and due to passing the information through the randomized response [18].

4.4.2 Randomized response implementation

The implementation of the randomized response [18] algorithm is a function consisting of a few simple conditions. Firstly, a coin toss is performed. This is done using Python’s module for generating pseudo-random numbers. The randomized response function uses the coin toss function to generate a number between 1 and 100 and returns either true if the number is greater or equal to 50 and false otherwise. If the first coin toss is true, it just returns the data. If false, it tosses the coin again. The second coin toss decides whether to report if the incident happened or not. If the toss is true, the *occurred* data entry is set to 1 and if false it is set to 0. This is done without taking into account if the *occurred* data entry was 0 or 1 to begin with. This realizes the randomized response algorithm and returns the desired result.

```
import math
import random

def coin_flip():
    r = random.randrange(1, 100, 1)
    if r <= 50:
        return True
    else:
        return False

def random_response(data):
    s = ""
    flip1 = coin_flip()
    if flip1:
        return data
    else:
        flip2 = coin_flip()
        data_list = data.split(",")
        v = data_list[4].split(":")
        if flip2:
            v[1] = ":1"
        else:
            v[1] = ":0"
        data_list[4] = v[0] + v[1]
        for i in range(len(data_list)):
            s += data_list[i] + ","
        return s[:-1]
```

The code above includes a software-based implementation of the randomized response [18], realizing the visualized version of the algorithm seen in Figure 3. The code also includes some code for rewriting the *occured* parameter mentioned in the previous section 4.4.1 to update the information.

4.4.3 Report formatting

The reporting system formats the report into an Excel (.xlsx) document using the Python library openpyxl [27]. If it is the first time an offense is being filed, there should exist no document and the program creates one. When an offense is to be reported, the program adds this row into the Excel document. With the new row added, the program takes the data and creates a chart in the Excel document which gives an overview of the offense's occurrence.

5 Evaluation Environment and Plan

To evaluate and test the presented solutions, different evaluation environments were used. This section of the report describes how the environments were implemented and integrated.

5.1 Gazebo

Gazebo is a 3d robotics simulator [28]. Gazebo features physics simulation, visualization, and integration with ROS. This project uses Gazebo and ROS to simulate a test environment for testing and presenting the system. The gazebo simulation node constantly posts information like the position of a vehicle which can be read by other subsystems like the Risk estimator. Gazebo, therefore, makes it possible to execute the system in a virtual environment.

5.2 Simulation environment

During the project, a test environment was developed for testing and demonstrating the risk detection subsystem and the system with ROS message passing as input. The system was developed using python and Gazebo. It includes both a test course, tools to create traffic scenarios in a four-way intersection and an interface.

The test course consists of a four-way intersection. With tools included in the environment, a traffic scenario can be scripted. These scenarios can include multiple vehicles driving at different speeds and acting in different ways in the environment. For instance, stopping before the intersection and then turning left. Furthermore, the simulation environment can be scripted to carry out multiple scenarios after each other to enable automatic testing. The results of these tests will then be automatically analyzed for relevant factors. For instance, when testing the risk estimator the collision horizon is of interest and that time delta is detected automatically.

5.3 Lab environment

For testing and demonstration purposes a physical lab environment was used. This environment includes a test track (Figure 8) and several Volvo trucks (Volvo Tamiya FH with a trailer, Figure 9) equipped with both a camera (Figure 10) and a Raspberry PI3 computer. The lab environment was used to collect videos of pre-planned traffic violation scenarios with two trucks driving simultaneously. This section will describe the different parts of the environment and illustrate how it was used.

5.3.1 Physical test track

The test track (Figure 8) is placed on a flat surface consisting of a small road network with turns, intersections, and a roundabout. Unlike in a real-life scenario, there are no external factors in this system such as pedestrians or animals. The road markings in the track include:

- Broken white lines - indicates that it is change lane to overtake a vehicle ahead, but with caution
- Solid white lines - separates the road from the rest of the test area
- Stop line - is drawn perpendicular to the road and implies that the driver should stop before continuing
- Give way - triangular road markings perpendicular to the road that implies that the driver should give way for other drivers before continuing

The test track is used for collecting videos of pre-planned traffic violation scenarios, to then be used for training the neural network in the image processing subsystem (see section 4.3).

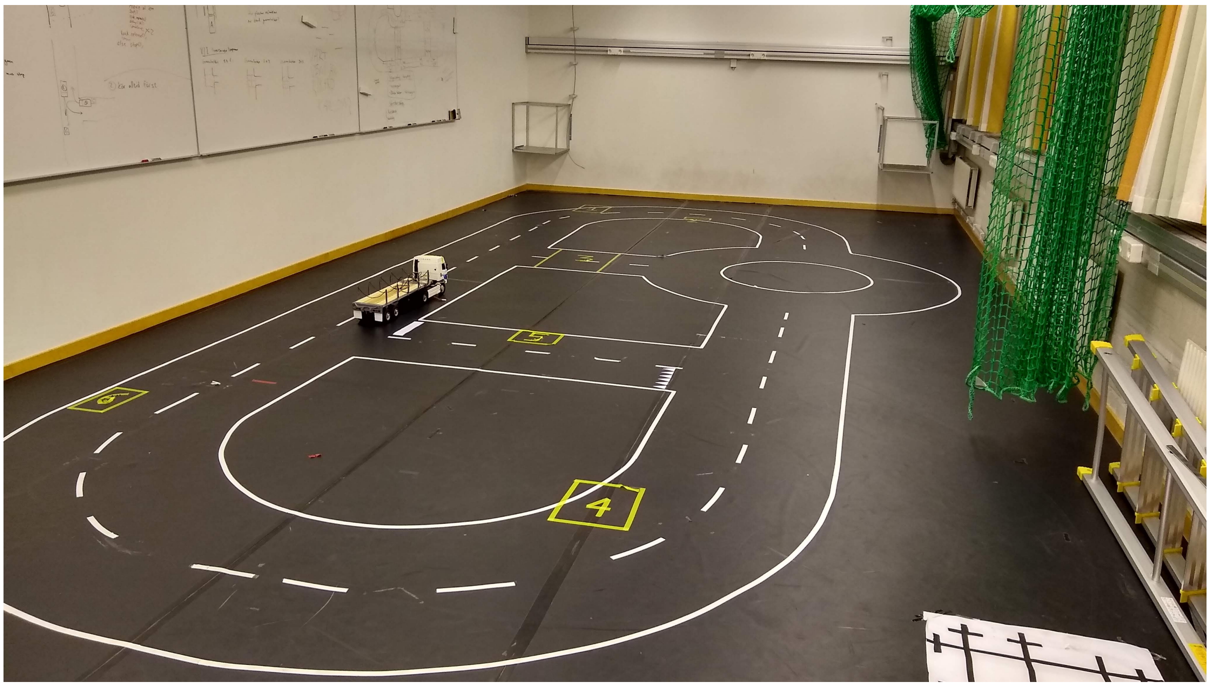


Figure 8: Physical test environment

We note that the testbed that we used was based on Gulliver testbed, which is described in [29]–[31]. The Gulliver initiative has lead to a number of developments in the area of vehicular positioning [32], cooperative driving [33]–[42] and communications [43]–[46].

5.3.2 Test vehicles

The vehicles used for testing and demonstrating are Volvo Tamiya FH trucks (Figure 9). Since the trucks are equipped with a trailer they will be more restricted in maneuverability, similar to full-size trucks.

The trucks are equipped with a Raspberry Pi 3, which is a 64-bit Single-Board Computer (SBC) with an ARM processor. This particular Raspberry Pi 3 is running the Linux based operating system Raspberry Jessie. The SBC is used to send signals for controlling the truck's engine and servo to then communicate to a laptop via ROS. Simple controllers are used to navigate the vehicles when collecting footage, demonstrating, and testing.

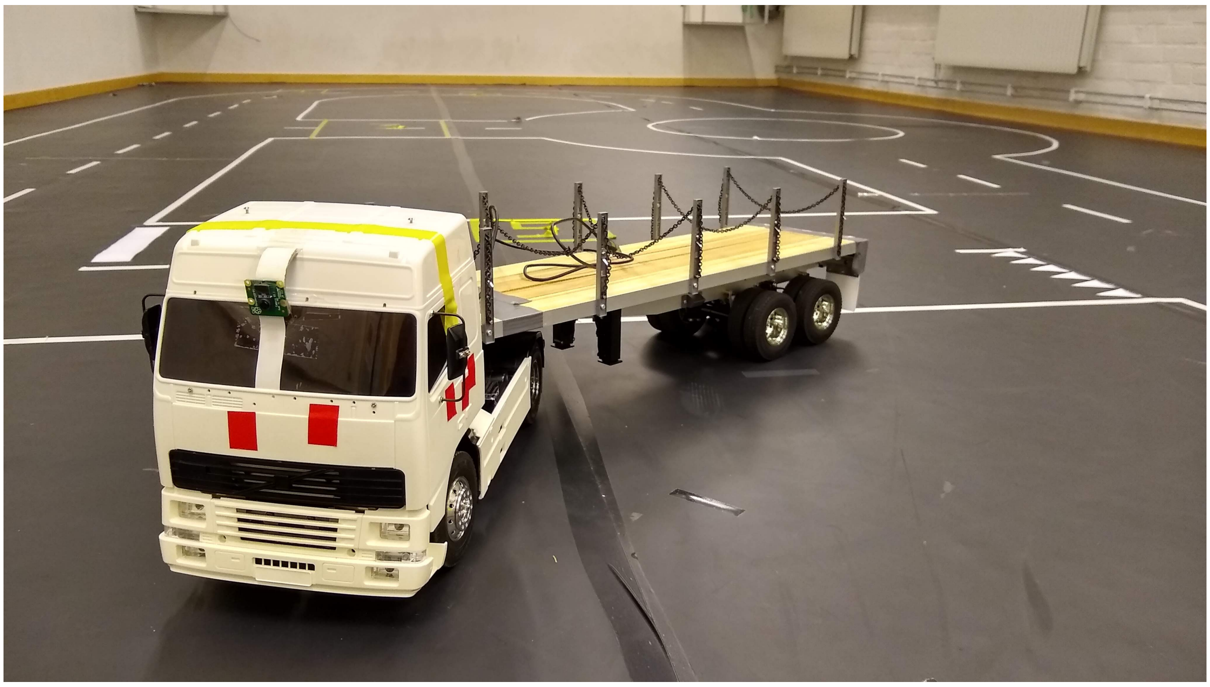


Figure 9: Volvo Tamiya FH truck

The camera used for collecting video footage is a Raspberry Pi-Camera V2 (Figure 10), which is capable of capturing videos in the resolution 1920x1080 with 30 frames per second (FPS) as well as 1280x720 with 60 FPS. The lower resolution but higher FPS setting were used to capture footage that was then to be processed in the image processing subsystem. Different scenarios were reenacted, *e.g.* crossing a solid white line.

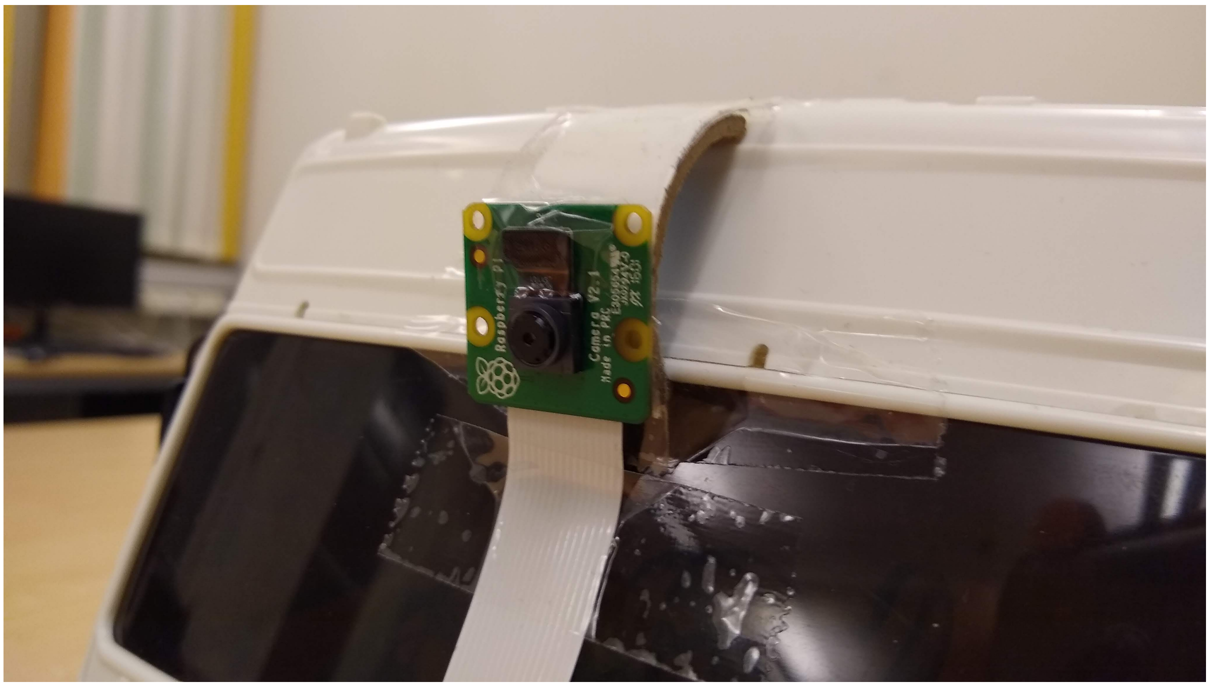


Figure 10: Raspberry PI-camera V2

5.4 Experiment plan

Evaluation is a crucial part of the development of a system. The following sections will walk through the thought-out plan to test each subsystem and the integrated system.

5.4.1 Image processing

The image processing subsystem will be tested in stages. Firstly we will test the efficiency and accuracy of object detection by our computer vision neural net. This is a necessary precondition for and used in conjunction with our traffic rule violation algorithm. When these results are acceptable we will move on to testing the traffic rule violation scenario detection algorithms. We will test both quantitative metrics such as confidence in object detection as well as system reliability in accurately detecting traffic rule violation scenarios without false positives.

1. Object Detection - We will perform thorough testing of the performance and accuracy of our trained computer vision neural net. Its ability to correctly identify the trained classes in recorded data not part of the training set. Our main focus is detection with high confidence and the ability to detect multiple object classes in close proximity correctly. Class objects that are obscured or in sub-optimal angles will also be analyzed as this is crucial in a high-density traffic situation. The systems' computing performance will also be considered as a supplementary metric as this affects the systems used in real-time detection.
2. Traffic Violation Detection - The ability of our algorithm to detect scenarios will be tested by presenting the system with test footage. This test footage will contain

segments where we perform traffic rule violations and segments with lawful driving. Evaluation will be done on both a pass/fail basis as well as reliability to produce these results consistently.

Multiple passes will be made with each set of test footage. Ensuring that adequate confidence in object detection and proper identification of traffic rule violation scenarios are met. Should the system perform lower than a preferred threshold with these metrics issues will be documented. Should we deem it necessary further improvements to the system will then be made depending on the issue, additional neural net training, or reworking scenario detection algorithms.

5.4.2 Risk detection

The risk detection subsystem will be tested with simulated traffic scenarios. The test is based on methods used previously to test the risk estimator [7]. However, unlike previous testing the traffic scenarios will be simulated in Gazebo which communicates with The risk detection subsystem via ROS and therefore emulates the real system. The tests will be run in a four-way give way intersection using two cars. One car has priority and the other one needs to give way. The following four scenarios will be evaluated:

1. The other vehicle drives in the give-way lane and then merges right in to the lane of the priority vehicle.
2. The other vehicle drives in the give-way lane and then merges left in to the opposing lane of the priority vehicle.
3. The other vehicle drives in the give-way lane and then crosses the lane of the priority vehicle.
4. The other vehicle drives in the opposing lane of the priority vehicle and then crosses the priority vehicle lane by turning left.

Each scenario is run in 12 test cases. In half the cases traffic rules are followed and in the other half they are not, which causes a collision. In each test case the speed and start position of each vehicle varies. Each test case is run two times to average out deviation so a total of 96 tests are performed. The scenarios have been used previously to evaluate the risk estimator. However, in theses test the risk estimator will be evaluated as a part of the risk detection system. The scenarios were chosen because they are the most common causes of traffic accidents on a four-way give way intersection. The scenarios are illustrated in Figure 11. The risk detection subsystem will be evaluated in the following ways.

The test evaluates the following variables:

1. How many false alarms are reported, namely a safe test case was classified as dangerous by the risk estimator.
2. How many dangerous situations were missed, namely an unsafe test wasn't classified as dangeous.
3. How much time has passed since a situation was classified as dangerous until the

collision occurs (The collision horizon). This collision horizon can be calculated by subtracting the time of the first classification of a dangerous situation with the time of collision.

Furthermore, the results will be compared by scenario and also results from previous reports.

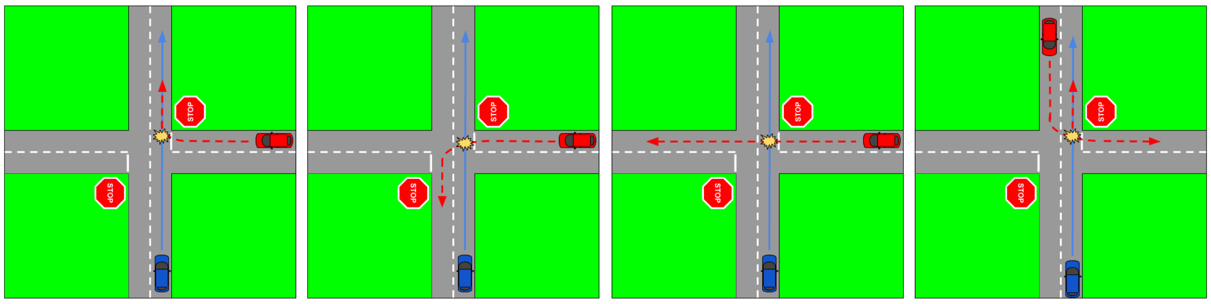


Figure 11: Simulated collision scenarios

5.4.3 Privacy preserving reporting

The testing of the privacy reporting subsystem is an important step in the development of the whole system as it reports all the data produced by the Traffic Violation Detection and Risk Estimator. Therefore, assessing the accuracy of the randomized response algorithm is crucial. The goal of the randomized response algorithm is to return and document the event occurrences without compromising individuals' privacy. But it is also important that the data is accurate as to be usable in determining if any action is needed to be taken.

Testing the implementation of the randomized response algorithm is done by running a test script. The script prompts the user to input the number of times the script should run the algorithm and also a probability value that the coin toss will return the value "True". When running the script, it takes a predefined test string of data, with the *occurred* variable set to 1, to run the randomized response algorithm with. The data string is interpreted as a traffic violation or a risky situation. The script then simply loops the algorithm with the test data for n amount of times and then prints out the average of the *occurred* variable as well as the standard deviation.

To get an overview of how the algorithm behaves and to assure that it is correctly implemented, the script should run a number of times with different values of n, *i.e.* 20, 50, 100, 500, 1000, 2000. The algorithm's coin toss probability will also be tested with values ranging from $0.5 - 0.70$. The test result when using a smaller sample size is expected to differ from a larger sample size. As the sample size grows, the average of *occurred* should be more consistent for each run.

6 Results

In this section, the results of the tests performed on the different subsystems will be demonstrated.

6.1 Image processing

As mentioned earlier, running real-time computer vision object detection in a resolution and bit rate adequate for use in traffic situations proved to demand quite high compute performance. The result of our research in image processing led us to understand that we needed highly specialized weights in our Neural Net detection system. Limiting training to only class objects absolutely necessary in deducing the traffic scenarios associated with traffic rule violations, in order to minimize compute cost.

YOLO runs on both the GPU and CPU and it's a heavy program to run. We evaluated different hardware and ran different stress tests on the CPU and GPU to check which bitrate, resolution, and framerate were the most suitable for this project. In Table 2 the results of different bit rates, resolutions, and GPUs are shown. We realized that the Jetson TX2 was too weak to realistically train and validate a Neural Network and had to use a more powerful computer. We concluded that both an NVIDIA 2080ti and NVIDIA 1060 are sufficient for the scope of this project, reaching above 15 frames per second (fps) in real-time object detection on 720p with 6326 kb/s bitrate which we deemed fast enough to realistically detect live traffic violations.

Weight	Encoding Bitrate	Res Frames	Jetson TX2	NVIDIA 2080ti (CPU Bottleneck)	NVIDIA 1060	NVIDIA 1660 super
YOLOV3	h264 4329 kb/s	1080p 30fps	1.5fps	22fps	5.5fps	x
YOLOV3	h264 12589 kb/s	1080p 60fps	1.5fps	22fps	5.3fps	x
YOLOV3	h264 6326 kb/s	720p 60fps	x	24fps	5.1fps	x
YOLOV3_tiny	h264 4329 kb/s	1080p 30fps	8fps	26fps	15.2fps	x
YOLOV3_tiny	h264 12589 kb/s	1080p 60fps	8fps	26fps	15.7fps	x
YOLOV3_tiny	h264 6326 kb/s	720p 60fps	x	60fps	14fps	x

Table 2: GPU hardware benchmarks for different bitrates running YOLO.

Different resolutions and bitrates were also tested to observe the CPU load, as shown in Table 3. It was clear that higher bitrates and resolutions directly affected the CPU load, and had less effect on the GPU. The stress tests clearly showed that the NVIDIA 2080ti was bottlenecked by the CPU since it never reached full load during the stress tests. The NVIDIA 2080ti load while using the less accurate Neural Network weight "YOLOV3_tiny" only used 16-27% of its computing power while the CPU load ranged from 82.9%-98.3%. This is because YOLO converts images into a pre-set resolution before running its Object Detection Algorithms.

Weight	Encoding Bitrate	Res Frames	Frames	NVIDIA 2080ti	CPU
YOLOV3	h264 4329 kb/s	1080p 30fps	22fps	64% 187W 1243MB	93,7%
YOLOV3	h264 12589 kb/s	1080p 60fps	22fps	85% 185W 8095MB	92,6%
YOLOV3	h264 6326 kb/s	720p 60fps	24fps	73% 214W 8096MB	57,9%
YOLOV3_tiny	h264 4329 kb/s	1080p 30fps	26fps	16% 69W 1253MB	82,9%
YOLOV3_tiny	h264 12589 kb/s	1080p 60fps	26fps	17% 59W 1253MiB	98,3%
YOLOV3_tiny	h264 6326 kb/s	720p 60fps	60fps	27% 63W 1243MB	80,6%

Table 3: CPU hardware benchmarks for different bitrates running YOLO.

The results of the final Neural Network was overall good. We noticed that the camera on-board the test vehicles were sometimes unstable, resulting in a distorted frame. Figure 12 shows a print screen of a frozen frame during one of the tests runs. The Neural Network accurately detected Inner Road Markings (depicted I), Outer Solid Road Markings (depicted O), and Trucks (depicted T). The Neural Network worked well in videos from angles not previously trained on as well, but not as reliable as the angles the Neural Network was trained in.

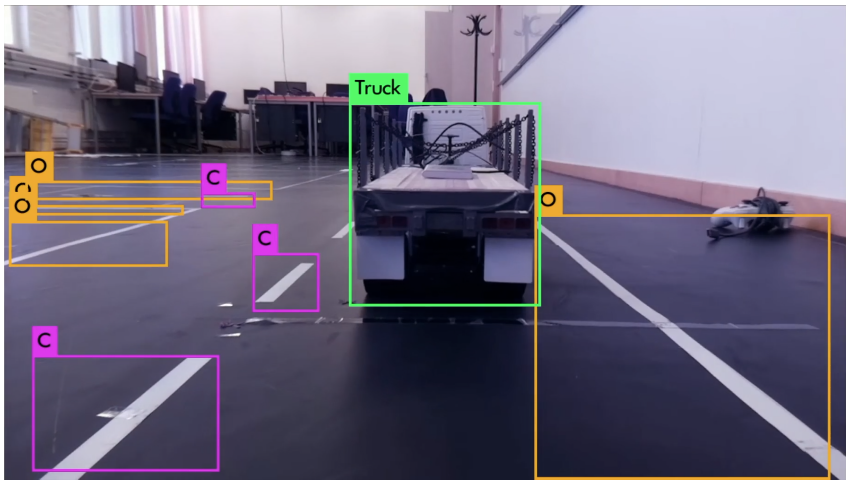


Figure 12: The Neural Network Detecting road markings

The final Neural Network was able to determine if a Truck had crossed over an outer solid line, as shown in the difference between Figure 13 a, b and c.

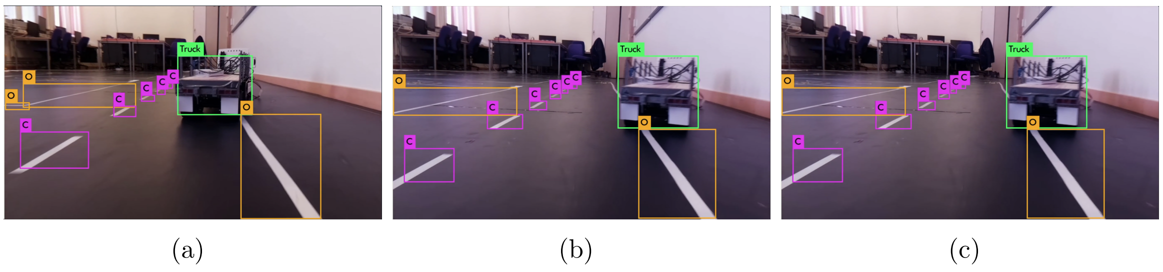


Figure 13: The Neural Network Detecting traffic violation

6.2 Risk detection

During the 48 non-dangerous traffic scenarios there were 16 false alarms, which is not ideal. However, there were no missed detect collisions during the 48 dangerous traffic scenarios. In the 15 the remaining test cases of undetected collisions as a function of the time until collision can be viewed.

The graph shows that the first detection occurred 7.3 s before collision. The average collision prediction horizon was 1.2 s, 0.5 s before collision 80 % of collisions were detected and 0.3 s all collisions were detected. This is comparable with results from a previous report in which the average prediction horizon was 2.5 s, The time of 80 % detection was 2 s and all collisions were detected at 0.5 [7]. As previous reports have stated this time span opens up to danger mitigation which requires 2-3s. For instance, an automatic braking system as proposed by this project [7].

The collision prediction horizon varied greatly depending on the traffic scenario as can be seen in Figure 14. Since the collision horizon is dependant on the performed maneuver, the results for the four different maneuvers are presented and compared in Figure 14. As seen in the graph, 100% of the collisions where detected at least 0.4 seconds before they occurred for all maneuvers except for crossing the path. For this maneuver, the average collision horizon was 0.58 seconds as opposed to 1.2 seconds for all scenarios combined. These results show that crossing the path is the most difficult to predict for the algorithm.

With these results, it can be concluded that the risk estimator was successfully integrated into the risk detection subsystem.

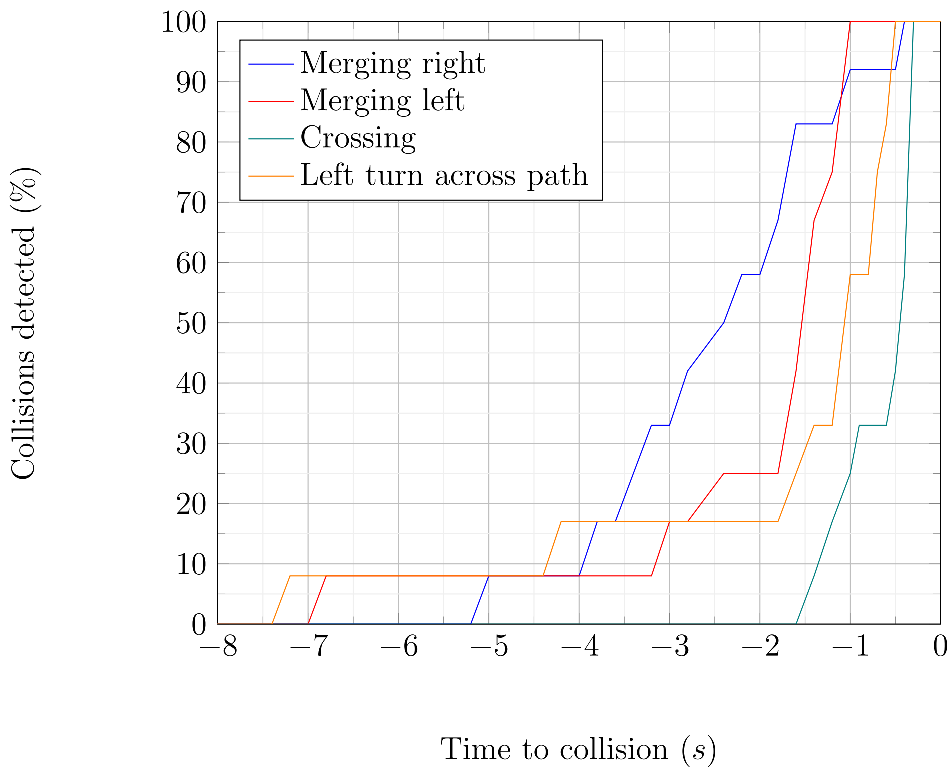


Figure 14: Percent of dangerous situations identified, Split by scenario

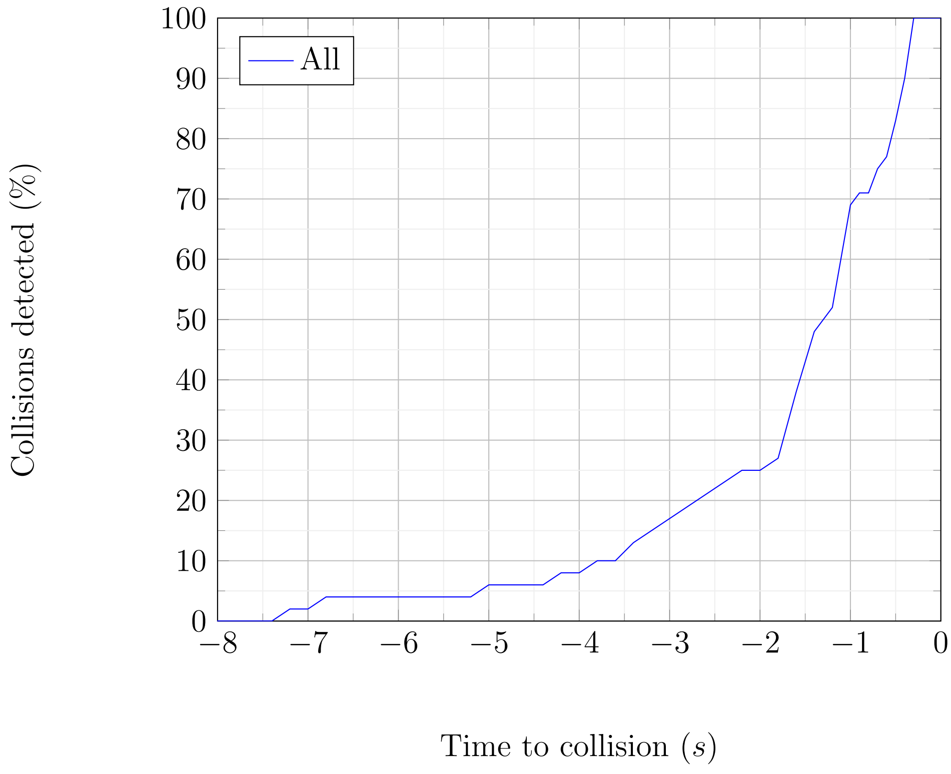


Figure 15: Percent of dangerous situations identified

6.3 Privacy preserving reporting

Whether the module is used by the image processing or the risk detection subsystem, the randomized response implementation functions as intended. The test script as described in section 5.4.4, gives the average value of the *occurred* variable (which is set to 1 in the sample string) with respect to a given sample size n and coin toss probability P. Results of the test with different sample sizes vary as shown in table 4. With a growing n, the *occurred* average converges to roughly 0.75 with $P = 0.50$. Increasing the coin toss probability results in a smaller deviation and thereby a more compact distribution of the *occurred* average.

The average and the standard deviation of *occurred* shown in the table is itself an average of 20 runs for each combination of sample size and coin toss probability.

n	$P = 0.50$ average/deviation	$P = 0.60$ average/deviation	$P = 0.70$ average/deviation
20	0.760 / 0.395	0.832 / 0.351	0.913 / 0.244
50	0.762 / 0.421	0.865 / 0.334	0.916 / 0.267
100	0.767 / 0.419	0.846 / 0.357	0.916 / 0.272
500	0.754 / 0.431	0.847 / 0.359	0.912 / 0.282
1000	0.759 / 0.428	0.844 / 0.363	0.915 / 0.279
2000	0.755 / 0.430	0.847 / 0.360	0.915 / 0.278

Table 4: Results from testing the randomized response implementation with different sample sizes (n) and coin toss probabilities (P).

These results show (see table 4) that the randomized response implementation gains accuracy in the average value of *occurred* and precision as the standard deviation is reduced when the probability of P is increased. Furthermore, there is less variability in the average value over the range of sample sizes tested for when $P = 0.70$. For all probabilities tested, the results converged when the sample size increased. Though the probability of $P = 0.70$ converged the fastest making it the optimal choice for use, it is possible that all scenarios will not necessarily reach such large sample sizes in actual traffic environments.

7 Discussion

This chapter will analyze and discuss the gathered results to evaluate the functionality of the system.

7.1 Image processing

Some work had to be put into sourcing the computer hardware needed to perform the highly taxing, in both computing power and time allotted, training of our Neural Network. We got approval for a purchase order for an NVIDIA 1660 Super graphics card. This was to be installed on a campus computer and thereafter be at our disposal during the project. These plans were however unfortunately stunted by the outbreak of the COVID-19 pandemic. Following this, the project group transitioned to remote work and relied solely on personal hardware, including an excessive NVIDIA 2080ti which is the GPU used in the remote training tool server. This situation led to some set-backs and challenges in completing the training of our Neural Network. It also led to some innovative development of Cloud-based remote work tools (see section 4.3.1) that can be used in the many tasks involved in the Neural Network training.

COVID-19 also limited the access to the lab environment at Chalmers, which in turn made it harder for us to collect images and video footage. The lack of footage made it harder to train and validate the Neural Network for different traffic rule violations, like stoplights and intersections give-way. Given the situation, we were able to produce a good product and the YOLO Neural Network was able to accurately detect several core traffic objects, like center lines and solid lines. Given different circumstances, we feel that the final product would have included more violations to detect.

We did not consider an exact technological solution to implement our prototype system in real-world vehicles. This could be done by developing a custom-built system or using ready purpose-built systems such as NVIDIA's DRIVE AGX line of products [47].

7.2 Risk detection

The results show that the risk estimator has a high success rate of predicting collisions early enough to deter from the crash. As seen in the results, 80% of the collisions were predicted at a minimum of 0.5 seconds before the crash. This time span could be enough to activate an autonomous braking system which would decrease the severity of, if not completely prevent, these collisions. Additionally the test results demonstrate that IARE can be used effectively with ROS. Something that has to be taken into account however, is the fact that we had a fairly high percentage of false alarms. This is potentially because of the inadequate accuracy of the tests in representing a real traffic situation. If the tests were created using data from real traffic situations the false positives could be fewer. Another way to decrease the number of false alarms would be to increase the threshold for collision detection. This would however, with a high probability, lead to a decrease in time between collision and predicted collision.

As mentioned before, the collision threshold could be adjusted to get less false alarms, at the cost of prediction time. How high the threshold should be depend on what we want to get out of the system. If the sole purpose of the system is to detect and report collisions the threshold would be set very high since it doesn't matter how long before the collision we detect the risk. However, if the system is also to be used to notify drivers that they are approaching a dangerous situation the threshold needs to be lower so the driver has time to act on the warning. In this project we wanted the latter so we prioritized time between detection and collision, resulting in more false alarms.

7.3 Privacy preserving reporting

As the results show, it is evident that the randomized response algorithm is a reliable approach to detach an individual from a given set of data entries. From a software development standpoint, it is also clear that said algorithm is efficient yet fairly straightforward to implement in software and to test. Further optimizations could be made on the probability of the coin toss returning true or not. These optimizations would allow the randomized response algorithm to return more precise results, meaning that variation would be reduced. Thus an area in a traffic environment would require fewer violations to occur for the result becoming conclusive of the problems that might be prevalent.

7.4 Social and ethical aspects

The purpose of this project is to create methods and techniques that provide important traffic violation information to government agencies and a risk detection system without comprising civil liberties or privacy. In this section, we will reflect on the social and ethical considerations of this project.

7.4.1 Traffic safety

In 2019 there were 223 persons who died in a traffic accident in Sweden [48]. The Swedish Transport Agency has a vision that no person should be killed or seriously hurt in the traffic. A reduction in traffic accidents would not only save lives but would also save society money. According to the Swedish Civil Contingencies Agency, road traffic accidents cost Sweden 21 billion SEK in direct and indirect costs in 2005 [49]. The techniques developed in this project could help the Swedish transport agency or other government agencies gathering information about dangerous and illegal activities on the road. This information could then help agencies in precautionary measures, like speed cameras or lower speed limits, to make traffic safer. Furthermore, the risk detection system could be integrated with an automatic braking system to prevent accidents. For instance, it has been shown that low-speed automatic emergency braking technology led to a 38% reduction in real-world rear-end crashes [50].

7.4.2 Privacy and surveillance

Privacy is a hard concept to clearly define. Some believe it is a collective name for the results of multiple rights. Others believe that it is a unified right in itself. Several benefits of privacy have been highlighted. One of them being that privacy gives the individual space where they have a greater degree of behavioral freedom. What is clear is that people value privacy. In a 2015 poll 74% of US adults said that controlling their private information was very important to them [51].

A new threat to privacy is the increased surveillance caused by digitization. According to a 2019 poll 46% of swedes felt concerned about large corporations like Google or Facebook threatening their privacy [51]. Surveillance could have multiple negative consequences. For instance, when large amounts of personal data are stored a data breach could give private information to malicious actors who could cause harm. Up to 35% of known identity thefts in the US are caused by corporate data breaches [52]. Another negative consequence would be the loss of privacy which could have harmful psychological effects. Increasingly connected vehicles will present a challenge to preserving privacy. However, this development could be prevented by the randomized response reporting techniques used in this project.

8 Conclusions

The project has successfully implemented a system that reports traffic violations and related road risks, without compromising individual integrity. These different capabilities were divided into three subsystems. The first subsystem detected traffic law violations using image processing techniques such as YOLO. The second subsystem identified risky situations in simulated intersections using Intention-aware risk estimation. The third subsystem reported these risks and law violations without compromising integrity using randomized response reporting. Furthermore the subsystems' capabilities were proved to work using an number of different tests. The project goal of creating a system prototype for improving traffic safety without compromising privacy was therefore successfully met.

8.1 Further development

Further development could be to integrate the system into a vehicle. Furthermore, it could increase the traffic environments the risk detection system would be compatible with. An additional future development we identified while working on the project is automated road signage improvement suggestions. As our system in its current form both use computer vision to identify road signage currently in place as well as road situations and ability to flag for dangerous traffic situations. We feel that a natural progression could be to identify where the road situation could be improved by proper road signage, should it not be present.

References

[1] N. H. T. S. Administration. (2018). Newer cars are safer cars, [Online]. Available: https://www.nhtsa.gov/newer-cars-are-safer-cars (visited on 02/02/2020).

[2] ——, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey", 2018, pp. 1–2. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812506 (visited on 02/02/2020).

[3] C.-Y. Chan, *Advancements, prospects, and impacts of automated drivingsystems*, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2046043017300035 (visited on 02/12/2020).

[4] TrafikAnalys, "Uppkopplade, samverkande och automatiserade fordon, farkoster och system – ett kunskapsunderlag", Swedish, 2019, pp. 23–38. [Online]. Available: https://www.trafa.se/globalassets/rapporter/2019/rapport-2019_8-uppkopplade-samverkande-och-automatiserade-fordon-farkoster-och-system---ett-kunskapsunderlag.pdf (visited on 02/03/2019).

[5] ——, "Uppkopplade, samverkande och automatiserade fordon, farkoster och system – ett kunskapsunderlag", Swedish, 2019, p. 79. [Online]. Available: https://www.trafa.se/globalassets/rapporter/2019/rapport-2019_8-uppkopplade-samverkande-och-automatiserade-fordon-farkoster-och-system---ett-kunskapsunderlag.pdf (visited on 02/03/2019).

[6] Datainspektionen. (2019). Fines and warnings. Swedish, [Online]. Available: https://www.datainspektionen.se/other-lang/in-english/the-general-data-protection-regulation-gdpr/fines-and-warnings/ (visited on 02/05/2019).

[7] J. I.-G. Stéphanie Lefèvre Christian Laugier, "Intention-aware risk estimation for general traffic situations, and application to intersection safety.", *[Research Report] RR-8379, INRIA.*, 2013, ffhal-00875356f.

[8] D. H. B. C. M. Brown, *Computer Vision.* Prentice Hall, 1982, ISBN: 9780131653160.

[9] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[10] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification", in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", *University of Washington*, 2018. [Online]. Available: https://arxiv.org/abs/1804.02767 (visited on 04/28/2020).

[13] J. Redmon, *Darknet: Open source neural networks in c*, http://pjreddie.com/darknet/, 2013–2016.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", *University of Washington, Allen Institute for AI, Facebook AI Research*, 2016. [Online]. Available: https://arxiv.org/abs/1506.02640 (visited on 04/28/2020).

[15] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger”, *University of Washington, Allen Institute for AI*, 2016. [Online]. Available: `https://arxiv.org/abs/1612.08242` (visited on 04/28/2020).

[16] S. Ammoun and F. Nashashibi, “Real time trajectory prediction for collision risk estimation between vehicles.”, *proc. IEEE Intelligent Computer Communication and Processing*, pp. 417–422, 2009.

[17] A. Eidehall and L. Petersson, “Statistical threat assessment for general road scenes using monte carlo sampling”, *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 137–147, 2008.

[18] S. L. Warner, “Randomized response: A survey technique for eliminating evasive answer bias”, *Journal of the American Statistical Association*, pp. 63–69, 1965. DOI: `10.2307/2283137`. [Online]. Available: `https://www.jstor.org/stable/2283137`.

[19] E. E. Khaled and D. Fida Kamal, “Protecting privacy using k-anonymity”, *J Am Med Inform Assoc.*, pp. 627–637, 2008. DOI: `10.1197/jamia.M2716`.

[20] Git, `https://github.com/`, 2020.

[21] Bitbucket, `https://bitbucket.org/`, 2020.

[22] O. Foundation, *Nodejs*, `https://nodejs.org/en/about/`, 2009–2020.

[23] E. You, *Vuejs framework*, `https://vuejs.org/`, 2014–2020.

[24] Google, *Firebase hosting*, `https://firebase.google.com/docs/hosting`, 2020.

[25] Wiki, *Ros (robot operating system)*, `https://www.ros.org/core-components/`, 2020.

[26] p. AlexeyAB, *Darknet github*, `https://github.com/AlexeyAB/darknet`, 2020.

[27] E. Gazoni, *Openpyxl*, `https://bitbucket.org/openpyxl/openpyxl/src/default/`, 2020.

[28] Gazebosim, *Gazebo*, `http://gazebosim.org/`, 2020.

[29] M. Pahlavan, M. Papatriantafilou, and E. M. Schiller, “Gulliver: A test-bed for developing, demonstrating and prototyping vehicular systems”, in *Proceedings of the 75th IEEE Vehicular Technology Conference, VTC Spring 2012, Yokohama, Japan, May 6-9, 2012*, IEEE, 2012, pp. 1–2, ISBN: 978-1-4673-0989-9. DOI: `10.1109/VETECS.2012.6239951`. [Online]. Available: `https://doi.org/10.1109/VETECS.2012.6239951`.

[30] C. Berger, E. Dahlgren, J. Grunden, D. Gunnarsson, N. Holtryd, A. Khazal, M. Mustafa, M. Papatriantafilou, E. M. Schiller, C. Steup, V. Swantesson, and P. Tsigas, “Bridging physical and digital traffic system simulations with the gulliver test-bed”, in *Communication Technologies for Vehicles, 5th International Workshop, Nets4Cars/Nets4Trains 2013, Villeneuve d’Ascq, France, May 14-15, 2013. Proceedings*, M. Berbineau, M. Jonsson, J. Bonnin, S. Cherkaoui, M. Aguado, C. R. Garcia, H. Ghannoum, R. Mehmood, and A. V. Vinel, Eds., ser. Lecture Notes in Computer Science, vol. 7865, Springer, 2013, pp. 169–184, ISBN: 978-3-642-37973-4. DOI: `10.1007/978-3-642-37974-1\_14`. [Online]. Available: `https://doi.org/10.1007/978-3-642-37974-1%5C_14`.

[31] C. Berger, O. M. Ponce, T. Petig, and E. M. Schiller, “Driving with confidence: Local dynamic maps that provide los for the gulliver test-bed”, in *Computer Safety, Reliability, and Security - SAFECOMP 2014 Workshops: ASCoMS, DECSoS, DEVVARTS, ISSE, ReSA4CI, SASSUR. Florence, Italy, September 8-9, 2014. Proceed-*

ings, A. Bondavalli, A. Ceccarelli, and F. Ortmeier, Eds., ser. Lecture Notes in Computer Science, vol. 8696, Springer, 2014, pp. 36–45, ISBN: 978-3-319-10556-7. DOI: 10.1007/978-3-319-10557-4_6. [Online]. Available: https://doi.org/10.1007/978-3-319-10557-4%5C_6.

[32] M. A. Skoglund, T. Petig, B. Vedder, H. Eriksson, and E. M. Schiller, "Static and dynamic performance evaluation of low-cost RTK GPS receivers", in *2016 IEEE Intelligent Vehicles Symposium, IV 2016, Gotenburg, Sweden, June 19-22, 2016*, IEEE, 2016, pp. 16–19, ISBN: 978-1-5090-1821-5. DOI: 10.1109/IVS.2016.7535357. [Online]. Available: https://doi.org/10.1109/IVS.2016.7535357.

[33] A. Casimiro, J. Kaiser, J. Karlsson, E. M. Schiller, P. Tsigas, P. Costa, J. Parizi, R. Johansson, and R. Librino, "Brief announcement: KARYON: towards safety kernels for cooperative vehicular systems", in *Stabilization, Safety, and Security of Distributed Systems - 14th International Symposium, SSS 2012, Toronto, Canada, October 1-4, 2012. Proceedings*, A. W. Richa and C. Scheideler, Eds., ser. Lecture Notes in Computer Science, vol. 7596, Springer, 2012, pp. 232–235, ISBN: 978-3-642-33535-8. DOI: 10.1007/978-3-642-33536-5_22. [Online]. Available: https://doi.org/10.1007/978-3-642-33536-5%5C_22.

[34] A. Casimiro, J. Kaiser, E. Schiller, P. Costa, J. Parizi, R. Johansson, and R. Librino, "The KARYON project: Predictable and safe coordination in cooperative vehicular systems", in *43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop, DSN Workshops 2013, Budapest, Hungary, June 24-27, 2013*, IEEE Computer Society, 2013, pp. 1–12, ISBN: 978-1-4799-0181-4. DOI: 10.1109/DSNW.2013.6615530. [Online]. Available: https://doi.org/10.1109/DSNW.2013.6615530.

[35] A. Casimiro, J. Rufino, R. C. Pinto, E. Vial, E. M. Schiller, O. M. Ponce, and T. Petig, "A kernel-based architecture for safe cooperative vehicular functions", in *Proceedings of the 9th IEEE International Symposium on Industrial Embedded Systems, SIES 2014, Pisa, Italy, June 18-20, 2014*, IEEE, 2014, pp. 228–237. DOI: 10.1109/SIES.2014.6871208. [Online]. Available: https://doi.org/10.1109/SIES.2014.6871208.

[36] O. M. Ponce, E. M. Schiller, and P. Falcone, "Cooperation with disagreement correction in the presence of communication failures", in *17th International IEEE Conference on Intelligent Transportation Systems, ITSC 2014, Qingdao, China, October 8-11, 2014*, IEEE, 2014, pp. 1105–1110, ISBN: 978-1-4799-6078-1. DOI: 10.1109/ITSC.2014.6957835. [Online]. Available: https://doi.org/10.1109/ITSC.2014.6957835.

[37] A. Casimiro, O. M. Ponce, T. Petig, and E. M. Schiller, "Vehicular coordination via a safety kernel in the gulliver test-bed", in *34th International Conference on Distributed Computing Systems Workshops (ICDCS 2014 Workshops), Madrid, Spain, June 30 - July 3, 2014*, IEEE Computer Society, 2014, pp. 167–176, ISBN: 978-1-4799-4181-0. DOI: 10.1109/ICDCSW.2014.25. [Online]. Available: https://doi.org/10.1109/ICDCSW.2014.25.

[38] V. Savic, E. M. Schiller, and M. Papatriantafilou, "Distributed algorithm for collision avoidance at road intersections in the presence of communication failures", in *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, IEEE, 2017, pp. 1005–1012, ISBN: 978-1-5090-4804-5. DOI: 10.1109/

IVS.2017.7995846. [Online]. Available: https://doi.org/10.1109/IVS.2017.7995846.

[39] T. Petig, E. M. Schiller, and J. Suomela, “Changing lanes on a highway”, in *18th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems, ATMOS 2018, August 23-24, 2018, Helsinki, Finland*, R. Borndörfer and S. Storandt, Eds., ser. OASICS, vol. 65, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 9:1–9:15, ISBN: 978-3-95977-096-5. DOI: 10.4230/OASIcs.ATMOS.2018.9. [Online]. Available: https://doi.org/10.4230/OASIcs.ATMOS.2018.9.

[40] O. M. Ponce, E. M. Schiller, and P. Falcone, “How to stop disagreeing and start cooperatingin the presence of asymmetric packet loss”, *Sensors*, vol. 18, no. 4, p. 1287, 2018. DOI: 10.3390/s18041287. [Online]. Available: https://doi.org/10.3390/s18041287.

[41] A. Casimiro, E. Ekenstedt, and E. M. Schiller, “Membership-based manoeuvre negotiation in autonomous and safety-critical vehicular systems”, *CoRR*, vol. abs/1906.04703, 2019. arXiv: 1906.04703. [Online]. Available: http://arxiv.org/abs/1906.04703.

[42] ——, “Self-stabilizing manoeuvre negotiation: The case of virtual traffic lights”, in *38th Symposium on Reliable Distributed Systems, SRDS 2019, Lyon, France, October 1-4, 2019*, IEEE, 2019, pp. 354–356, ISBN: 978-1-7281-4222-7. DOI: 10.1109/SRDS47363.2019.00048. [Online]. Available: https://doi.org/10.1109/SRDS47363.2019.00048.

[43] O. Landsiedel, T. Petig, and E. M. Schiller, “Dectdma: A decentralized-tdma - with link quality estimation for wsns”, in *Stabilization, Safety, and Security of Distributed Systems - 18th International Symposium, SSS 2016, Lyon, France, November 7-10, 2016, Proceedings*, B. Bonakdarpour and F. Petit, Eds., ser. Lecture Notes in Computer Science, vol. 10083, 2016, pp. 231–247, ISBN: 978-3-319-49258-2. DOI: 10.1007/978-3-319-49259-9_19. [Online]. Available: https://doi.org/10.1007/978-3-319-49259-9%5C_19.

[44] T. Petig, E. Schiller, and P. Tsigas, “Self-stabilizing TDMA algorithms for wireless ad-hoc networks without external reference”, in *13th Annual Mediterranean Ad Hoc Networking Workshop, MED-HOC-NET 2014, Piran, Slovenia, June 2-4, 2014*, IEEE, 2014, pp. 87–94, ISBN: 978-1-4799-5258-8. DOI: 10.1109/MedHocNet.2014.6849109. [Online]. Available: https://doi.org/10.1109/MedHocNet.2014.6849109.

[45] T. Petig, E. M. Schiller, and P. Tsigas, “Self-stabilizing TDMA algorithms for wireless ad-hoc networks without external reference”, in *Stabilization, Safety, and Security of Distributed Systems - 15th International Symposium, SSS 2013, Osaka, Japan, November 13-16, 2013. Proceedings*, T. Higashino, Y. Katayama, T. Masuzawa, M. Potop-Butucaru, and M. Yamashita, Eds., ser. Lecture Notes in Computer Science, vol. 8255, Springer, 2013, pp. 354–356, ISBN: 978-3-319-03088-3. DOI: 10.1007/978-3-319-03089-0_28. [Online]. Available: https://doi.org/10.1007/978-3-319-03089-0%5C_28.

[46] M. Mustafa, M. Papatriantafilou, E. M. Schiller, A. Tohidi, and P. Tsigas, “Autonomous TDMA alignment for vanets”, in *Proceedings of the 76th IEEE Vehicular Technology Conference, VTC Fall 2012, Quebec City, QC, Canada, September 3-6, 2012*, IEEE, 2012, pp. 1–5, ISBN: 978-1-4673-1880-8. DOI: 10.1109/VTCFall.

2012.6399373. [Online]. Available: https://doi.org/10.1109/VTCFall.2012.6399373.

[47] NVIDIA. (2020). Nvidia drive agx, [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/drive-platform/hardware/ (visited on 06/02/2020).

[48] Transportstyrelsen. (2020). Nationell olycksstatistik, [Online]. Available: https://www.transportstyrelsen.se/sv/vagtrafik/statistik/Olycksstatistik/officiell-statistik-polisrapporterad/nationell-statistik/ (visited on 02/07/2020).

[49] S. O. Linda Ryen, “Samhällets kostnader för vägtrafikolyckor”, Swedish, 2009, p. 111. [Online]. Available: https://rib.msb.se/filer/pdf/25603.pdf (visited on 02/07/2020).

[50] B. fields et al., “Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes”, *Accident Analysis Prevention*, 2015. DOI: 10.1016/j.aap.2015.03.029. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0001457515001116.

[51] I. Stiftelsen, “Svenskarna och internet 2019”, *Internet Stiftelsen*, 2019. DOI: ffhal-00875356f. [Online]. Available: https://svenskarnaochinternet.se/app/uploads/2019/10/svenskarna-och-internet-2019-a4.pdf.

[52] J. Research, “Identity fraud survey report”, *Javelin Research*, 2006. DOI: ffhal-00875356f.