TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

INSTITUT FÜR
MECHANIK UND
MECHATRONIK
Mechanics & Mechatronics

Diploma Thesis

A Context-Aware Grid-Based Trust Management Model applied to Collective Perception

carried out for the purpose of obtaining the academic degree of a master of science
under the supervision of

Univ.Prof. Dr. Stefan Jakubek
Institute of Mechanics and Mechatronics
Department of Control Engineering and Process Automation

and

Msc. João-Vitor Zacchi
Fraunhofer Institute of Cognitive Systems IKS

submitted to the Vienna University of Technology

Faculty of Mechanical Engineering and Management Sciences

from

Oliver Ecker
Matr.Nr.: 01629064

Declaration on oath

I declare in lieu of an oath that I have written this work independently, that I have not used any auxiliary materials other than those indicated, and that I have cited all formulations and concepts taken from unprinted sources, printed literature, or the Internet, either in wording or in essential content, in accordance with the guidelines for scientific papers, and that I have marked them with footnotes or identified them with exact references.

Vienna, April 17, 2023

Oliver Ecker

Acknowledgements

First of all, I would like to thank my supervisor Professor Stefan Jakubek for giving me the opportunity to write my thesis abroad and in collaboration with Fraunhofer IKS. It was a very special chance for me not only to see how other research institutes work, but also to work with colleagues with different backgrounds and approaches to problems. I appreciate your advice and ideas throughout the meetings, which ultimately helped me to improve my work. I also thank you for your help in obtaining the KUWI scholarship that allowed me to finance my stay abroad.

I would like to give special thanks to my co-supervisor João-Vitor Zacchi for his tireless support and co-supervision during the creation of my thesis. Thanks to our weekly meetings and countless brainstorming sessions, the thesis has become what it is today. I greatly appreciate you taking the time to help me generate the necessary data for the simulations and develop the object detection algorithm. Even though these parts are not the focus of this work, they play an important role in testing my algorithm.

My sincere thanks go to Núria Mata, who helped me organize the work and allowed me to work abroad several times. Her expertise in the automotive sector helped me a lot to understand what I should focus on in the report.

Finally, I would like to thank María, who supported me every day and helped me overcome difficult situations during the work. I am very grateful to have you in my life.

Lay Summary

Connected vehicles can share information with their surroundings to make better decisions. Their shared information is typically evaluated based on measurements, which are depending on weather conditions. This thesis looks at how bad weather affects the ability of vehicles to share information with their surroundings. It suggests a new approach to tell whether certain vehicles tend to make inaccurate measurements based on the impact of strong sun light. Simulation results show that taking this information into account makes vehicle evaluation more meaningful and prevents the use of data from affected vehicles.

Abstract

Collective perception is a promising technology in the field of autonomous driving. It enables vehicles to connect with their environment by sending and receiving information to help them make decisions. So-called trust management models, which have the purpose of evaluating connected vehicles or other entities based on their measurements, play an important role in this context. Among other challenges, the quality of the transmitted data depends on the quality of sensor measurements, which is particularly influenced by the weather. As a result, difficult weather conditions affect the transmitted information and can lead to incorrect decisions by the receiving vehicles.

This thesis investigates the impact of challenging weather conditions on collective perception and in particular on the rating of the impacted vehicles. It proposes a novel approach that determines whether certain vehicles tend to make wrong measurements based on their orientation relative to the sun. This should in turn affect their rating and make them to be considered less in the collective perception fusion process.

The model has been tested using the open-source simulator CARLA. The evaluation shows that considering weather leads to an overall improvement of detection performance. It is therefore a reasonable factor to consider in the evaluation process of connected vehicles and leads to more meaningful ratings.

Preface

This work is an original and intellectual product of the author, Oliver Ecker. It has been written in collaboration with the Fraunhofer Institute of Cognitive Systems IKS. The object detection algorithm in chapter 4 has been implemented by the co-supervisor João-Vitor Zacchi.

Contents

Abbrevations

$3GPP$	3rd Generation Partnership Project
AD	Autonomous Driving
AS	Autonomous System
AV	Autonomous Vehicle
$C-ITS$	Cooperative Intelligent Transportation System
$C-V2X$	Cellular Vehicle-to-everything
CA	Collective Awareness
CAM	Collective Awareness Message
CP	Collective Perception
CPM	Collective Perception Message
CPS	Collective Perception Service
$ETSI$	European Telecommunications Standards Institute
FLS	Fuzzy Logic System
FN	False Negative
FoV	Field of View
$GNSS$	Global Navigation Satellite System
HDR	High Dynamic Range
$hFoV$	horizontal Field of View
$IEEE$	Institute of Electrical and Electronics Engineers

IMU	Internal Measurement Unit
IoT	Internet of Things
IOU	Intersection over Union
ITS	Intelligent Transport System
LiDAR	Light Detection and Ranging
LTE	Long-Term Evolution
NHSTA	National Highway Traffic Safety Administration
NPC	Non Player Character
OCB	Outside the Context of Basic Service Set
OMNeT++	Objective Modular Network Testbed in C++
PDU	Protocol Data Unit
RADAR	Radio Detection and Ranging
RSU	Road-Side Unit
SAE	Society of Automotive Engineers
SUMO	Simulation of Urban Mobility
TN	True Negative
TP	True Positive
V2I	Vehicle-to-infrastructure
V2N	Vehicle-to-network
V2P	Vehicle-to-pedestrian
V2V	Vehicle-to-vehicle
V2X	Vehicle-to-everything
VANET	Vehicular ad-hoc network
VEINS	Vehicles in Network Simulation

$vFoV$	vertical Field of View
$YOLO$	You Only Look Once

Chapter 1

Introduction

Autonomous driving has become a very popular topic in recent years due to a multitude of hoped-for benefits such as increased energy efficiency and passenger comfort. For autonomous driving to work, a reasonably accurate profile of the environment must be established in order for the algorithm to plan maneuvers and make decisions. For this purpose, a new technology called Collective Perception has recently emerged [1]. It is based on the idea of sharing data measured by multiple vehicles in order to obtain a more accurate picture of the environment. It aims to enable vehicles to spot objects that may be obscured by an obstacle, which could drastically improve their decision-making and ultimately make them safer in day-to-day operations. Since data vehicles received from others is not directly known to be correct, transmitting vehicles should be rated in a certain way to avoid using biased or malicious measurements for maneuver planning. To address this issue, trust management models are introduced, which use a wide range of techniques to rate the vehicles performance or their sent data by assigning them with trust values [2].

In order for autonomous driving vehicles to perceive their surroundings, they are usually equipped with a variety of sensors such as cameras or LiDAR sensors, which ideally complement each other and compensate for their weaknesses [3, 4, 5]. Most of these sensors are known to be very sensitive to weather conditions such as fog, rain or excessive sunlight. Therefore, it is desirable to account for this type of contextual information when analyzing the received measurement data. Although many types of trust management models have been presented in the past, none of them considered the influence of weather when assigning a trust score to vehicles.

This work is proposing a novel approach to consider weather context for computing trust of sending vehicles. It utilizes Fuzzy Logic, a method that allows the consideration of human expertise and scores with its interpretability [6]. These benefits are particularly

important for models applied in the field of autonomous driving since safety is is a key part of making this technology become more suitable for the real world.

The thesis is structured as follows: First the main principles of collective perception and trust management models are examined. Then, state-of-the-art trust management models are shown to give an overview of the current model architectures. Subsequently, the novel algorithm is proposed and explained in detail. Finally, the algorithm will be tested by performing a simulation study and by comparing the model performance for different scenario cases.

Chapter 2

Background

2.1 Autonomous Driving

Autonomous systems (AS) are becoming increasingly popular and have already conquered many areas of our world. They can be defined as systems that can change their behavior in response to unanticipated events during their operation [7]. This fundamentally distinguishes them from classical automated systems, where predicted environments are required for their usage and where the system is usually restricted in what tasks it can perform. AS are particularly advantageous in remote environments where human interaction is not possible or in hostile environments where it is just too dangerous for humans to interact and work [8]. Moreover, for activities that are very long and repetitive, AS can play a significant role in achieving better results than humans while working faster and more efficiently.
For AS to truly act without human intervention, they must be able to handle complex scenarios that may not even have been foreseeable at the time of their development. This is why they are usually defined using artificial intelligence to model and classify the complexity of real world environments with high accuracy [9].
There are numerous application areas of AS [10, 11, 12, 13], whereas this work addresses Autonomous Driving (AD). Autonomous Vehicles (AV's) equipped with AD technology relieve the burden on human drivers by performing various tasks such as obstacle avoidance, road sign recognition or lane departure warning. In addition, AD may enable the reduction of traffic congestion and lower emissions through the use of advanced fuel economy. Finally, AV's are designed to help people in their daily lives by providing a safe and reliable transportation option, e.g. for the elderly and disabled [14].

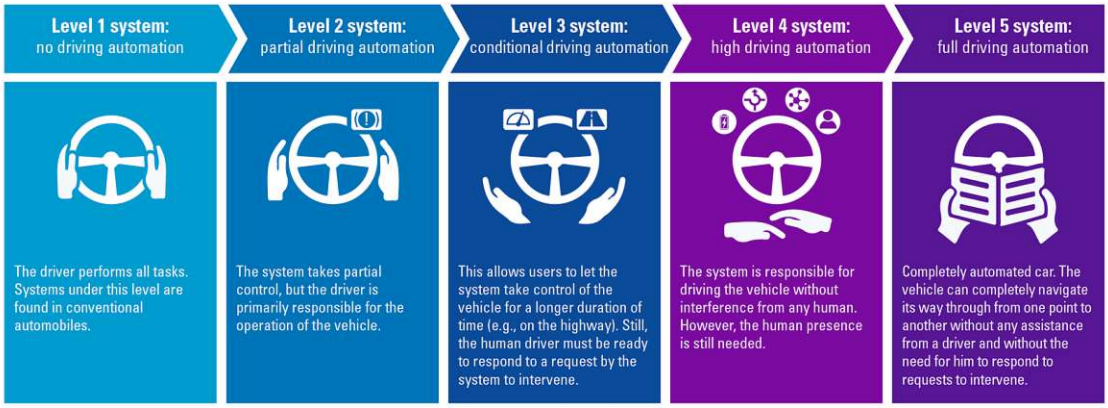

Figure 2.1: Overview on the SAE levels of driving automation [15, 16].

In general, AD can be divided into six levels defined by the Society of Automotive Engineers (SAE) [17]. Figure 2.1 shows the corresponding levels and a brief description. In summary, level 0 means no driving automation, level 1 limited driving support, and level 2 partial driving automation. Level 3 corresponds to conditional driving automation, level 4 to high driving automation, and level 5 to full driving automation. It should be noted that in the first three levels, the driver still has to perform all dynamic driving tasks himself, while in the last three levels the AD system takes over the corresponding tasks [18]. Since the system is not truly autonomous at the first three levels, the term AD is typically used for systems at level 3 and above.
Level 0 to 2 AD systems are already approved on the European market, while level 3 and 4 AD systems are currently being tested and are expected to be launched between 2020 and 2030. However, level 5 AD systems are not likely to enter the market before 2030 [19]. In both the U.S. and China, AD systems at Level 3 and above are not expected to be available before 2025 [15, 20].

2.1.1 Sensor types

In order for AV's to act independently, they need to measure their surroundings in the most accurate way, which requires them to have a sufficiently broad set of sensors. There are mainly four different sensor types installed in AV's, namely LiDAR, RADAR, Ultrasonic sensors, Cameras and GNSS sensors. In addition, IMU and odometry sensors are also installed in most AV's. However, these will not be discussed further in this work [3, 4, 5].

LiDAR

LiDAR (Light Detection and Ranging) sensors use lasers to create a point cloud of their environment with the aim of detecting and categorizing objects. Some typical applications are surveying, archaeology, robotics and last but not least AV's. They principally work by illuminating their field of view by a laser source while an array of photo-detectors at the image plane pick up time-of-flight information of each individual pixel. With this information, one is not only able to get the position of an object but also to calculate the distance. Despite their excellent ability to provide an accurate map of the environment, they have difficulty in certain weather conditions, as described in section 2.4.1, and are still quite expensive, although prices continue to drop for mass market use [21].

RADAR

RADAR (Radio Detection and Ranging) sensors transmit radio waves to their surroundings to detect the distance and speed of objects in the environment. They are mainly used to detect obstacles, as they are usually designed to identify any type of object. In contrast to LiDAR technology, RADAR sensors are more resistant to weather conditions. They are also quite inexpensive compared to other types of sensors, which is why they play a central role in AV's. Despite some other operational challenges such as interference and limited resolution, their main drawback is object classification, so they cannot be used exclusively for developing a fully autonomous system [3, 22].

Cameras

Cameras are important for the AV's to gather visual information about their environment, which can help them detect and classify objects based on their shape, color or infrared radiation. Usually, multiple cameras are installed in an AV to get a 360° view and not miss any object. The advantages of this type of sensor are that it is cheaper and easier to integrate than LiDAR, for example. However, there are also some disadvantages, such as their poor performance in bad lighting conditions, as this severely impairs their visibility and thus object detection and classification. In addition, they cannot provide information on depth as a primary metric, requiring it to be calculated separately, which can lead to imprecise estimates [23].

Ultrasonic sensors

Ultrasonic sensors emit high-frequency sound waves to their surroundings, which are reflected by the object and then processed by the internal system. The result is the

distance between the sensor and the corresponding object. Ultrasonic sensors can make a precise measurement within their sensing range, and their relatively low cost makes them user-friendly for many applications. In addition, their performance is not affected by weather or lightning, making them very attractive for AVs as well. However, due to their limited range and measurement rate, as well as their varying detection accuracy with different object materials, they cannot be used exclusively for AV's [24].

In table 2.1, the performance of LiDAR, RADAR, camera and ultrasonic sensors are compared. Their performance under specific weather and lightning conditions will further be discussed in section 2.4.1.

Feature	**LiDAR**	**RADAR**	**Camera**	**Ultrasonic sensor**
Primary Technology	Laser beam	Radio wave	Light	Sound wave
Range	200m	250m	200m	5m
Resolution	Good	Average	Very good	Poor
Detects distance	Good	Very good	Poor	Good
Cost	High	Low	Medium	Low
Size	Bulky	Small	Small	Small

Table 2.1: Summary of AV sensor performance [25].

GNSS position sensors

The Global Navigation Satellite System (GNSS) is a constellation of satellites that deliver signals from space as position and time data to GNSS sensors [26]. These sensors can then use this data to determine their location in terms of latitude, longitude, and altitude in real time, making them very suitable for AV's, where knowing the location of objects in the environment can greatly improve their detection and classification performance [27].

2.2 Vehicle-to-Everything and Collective Perception

Over the past years, a new technology called Internet of Things (IoT) has become one of the most important inventions of the 21st century. It describes a network that connects any objects that are equipped with software or sensors to its surroundings over the internet [28]. In this way, devices can exchange information as needed, allowing not only themselves but also other devices to adapt their interaction to their surroundings. IoT has many application fields [29, 30], though in this work the focus lies on Intelligent Transportation Systems (ITS). These systems are advanced applications that aim to provide innovative services related to different transport modes and traffic management, enabling users to be better informed and to use transport networks in a safer, more coordinated and "smarter" way [31]. A recent suggestion for the foundation of ITS are so-called Vehicular Ad-Hoc Networks (VANET's) [32].
Recently, Vehicle-to-Everything (V2X) communication got very popular in the field of ITS. According to [33], V2X refers to a set of standards and technologies that enables vehicles to cooperate with their current infrastructure, including road users like pedestrians and roadside units (RSU's). Through this collective sharing of information, a vehicle should be able to extend the perception beyond its Field of View (FoV), leading to the notion of Collective Perception (CP). The basic principle of CP is shown in the figure 2.2. Here it can be seen that vehicle 1 is not able to directly observe vehicle 3, while vehicle 2 is able to do so.

Figure 2.2: The basic principle of Collective Perception [34].

In order for vehicle 1 to extend its perception, vehicle 2 sends its measurement and

state information to vehicle 1 so that the latter can detect vehicle 3.

2.2.1 Advantages

V2X communication provides many benefits for ITS, especially in the field of AD. The most important of them are listed below.

Safety

According to the U.S. Department of Transportation's National Highway Traffic Safety Administration (NHTSA), 38824 people got killed in a traffic accident in 2020 in the U.S.. Even though the number of crashes decreased by 22% compared to 2019, the number of fatal crashes increased by 6%. In 45% of these fatal crashes, the drivers of the vehicles were engaged in at least one of the following risky behaviors: speeding, alcohol impairment, or not wearing a seat belt [35].
NHTSA believes that AD systems have a high life saving potential, since they are disconnecting the human driver from the chain of events that can lead to a crash. However, the implementation of AD is highly dependent on the vehicle's ability to perceive the environment. AV's require a certain level of knowledge about its environment in order to drive safely [36]. V2X communication has proven to be a promising technology to address this shortcoming by gathering helpful information such as sensor measurements from surrounding vehicles, road users or RSU's, allowing an AV to use CP in its favor and coordinate its actions [37].

Comfort

V2X communication reassures a sense of security. Vehicles with enhanced visibility through CP are also able to make driving in unfavorable conditions like poor visibility easier [38].

Cost Efficiency

V2X can help build a more efficient transportation system, which in turn saves money. Congestion and construction cause delays and can drive up the cost of doing business. V2X can help identify these obstacles and change the route of travel to save fuel and time [39].

Environmental Factors

V2X can help reduce environmental impact. Advanced applications such as platooning allow vehicles to follow each other at a very close distance while continuously interact-

ing with each other [40]. Not only does this have various safety or business benefits, but it is also believed that platooning can reduce fuel consumption and thus carbon dioxide emissions by up to 20%, depending on the driving scenario [41].

2.2.2 V2X Communication standards

An AV participating in the V2X communication network can interact with the environment in several ways, as shown in figure 2.3. With all these sources of information, each AV should get the information it needs to act in a safe and reliable manner and fulfill the objectives mentioned above.
Nowadays, two main communication standards are used to enable information exchange within the V2X network. These standards are briefly explained below.

IEEE standard

This standard, specified in 2010, is based on the IEEE 802.11p interface (IEEE 802.11 Outside the Context of Basic Service Set (OCB) mode), which operates in the 5.9 GHz band [42]. In Europe, the ETSI committee called an ITS system based on IEEE 802.11 OCB mode Intelligent Transport System G5 (ITS-G5), and the upper layer is denoted as Cooperative Intelligent Transportation System (C-ITS) [43, 44].
The target scenario of ITS-G5-based systems is short-range communication involving only vehicles and RSU's. This means, for these systems the term V2X refers to V2I and V2V only. There are several issues coming with ITS-G5-based systems as blocked transmission of specific frames or a lack of reliability and performance due to frame collisions and channel fading coming with higher traffic load [44].

3GPP standard

Starting in 2014, the 3rd Generation Partnership Project (3GPP) has been working on the standardization of in-vehicle communications based on the previously standardized 4G Long-Term Evolution (LTE) and later 5G mobile communications. This enables an operation not only in the 5.9-GHz band but also in the licensed bands of the cellular networks. For their systems, two different interfaces are used: A PC5 interface used for short-range data transmission and a Uu interface for long-range communication [44]. This technological approach relying on 4G LTE or 5G V2X communications is combined under the 3GPP standard for Cellular V2X (C-V2X) [45].
The C-V2X technology standard introduced two new communication paradigms, that significantly extend the range of interaction and use case scenarios: the involvement of the Cellular Mobile Network (V2N) directly connecting the vehicle with a ITS central

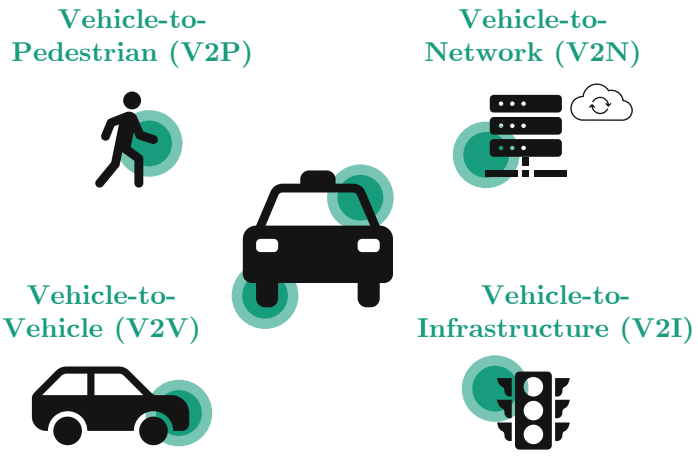

Figure 2.3: The figure above shows different ways in which an AV can interact with the environment.

system, and the involvement of pedestrians or cyclists through the use of mobile devices (V2P) [44]. According to [46], C-V2X systems can provide better performance than ITS-G5-based systems regarding latency, coverage length and throughput, especially in lower traffic densities.

2.3 Collective Perception Service

As mentioned in the previous sections, CP means the exchange of measurement and state information between vehicles (V2V) or other entities such as road users (V2P), infrastructure (V2I) or a mobile network (V2N). This distinguishes CP in particular from the concept of Cooperative Awareness (CA), where only information about the current state of the vehicle is exchanged. The concept of CA is not discussed further in this thesis; for more information, see [47]. The service providing CP within the V2X network is currently being standardized by ETSI and known as Collective Perception Service (CPS) [1].
In the following sections, the main components of the CPS are described and its specific message type is elaborated. In addition, use cases of CP are highlighted and redundancy mitigation techniques are explained.

2.3.1 Components

According to [48], any CPS can be split into three parts: Sensing, Communication, and Data Fusion. The corresponding components are further discussed in the following.

Sensing

This component includes the detection and classification of objects. One can divide these objects between static objects, such as buildings or trees, and dynamic objects as vehicles or pedestrians. Usually, static objects are obtained using map information, while dynamic objects have to be detected by sensors [48].
The raw data obtained from the sensors can either be transmitted directly or pre-processed first, the latter being recommended by ETSI, as raw data would place very high demands on data rates and transmission frequencies, which would further increase with the number of sensors attached to an entity [1].

Communication

The communication part deals with the transmission of data. As mentioned in section 2.2.2, there are two communication standards that entities can use: the IEEE standard used for ITS-G5 systems and the 3GPP standard used for C-V2X systems. The exchanged information is contained in CA messages and CP messages, the latter being the focus of this work and described in section 2.3.2.

Data Fusion

Entities in the CPS receive information from their own sensors as well as from other entities. All data must be pre-processed and fused accordingly to make it usable and thus enhance perception. There are mainly two different fusion approaches:

- Early Fusion: Multiple spatially diverse sensor measurements are combined before feature detection (fusion in feature space)
- Late Fusion: Sensor measurements are combined after feature detection (fusion in semantic space)

When to use early or late fusion schemes highly depends on the application field [49, 50], in chapter 3 common techniques for AV's are described.

2.3.2 Collective Perception Messages

The CPS uses so-called Collective Perception Messages (CPM's) for the transmission of measurement data and state information. These messages allow entities to transmit their sensor information as well as their state information to others.
A CPM currently consists of an ITS PDU header and five different types of containers: a Management Container, a Station Data Container, one or more Sensor Information Containers, one or more Perceived Object Containers and one or more Free Space Addendum Containers. The general CPM structure is shown in figure 2.4 [1].

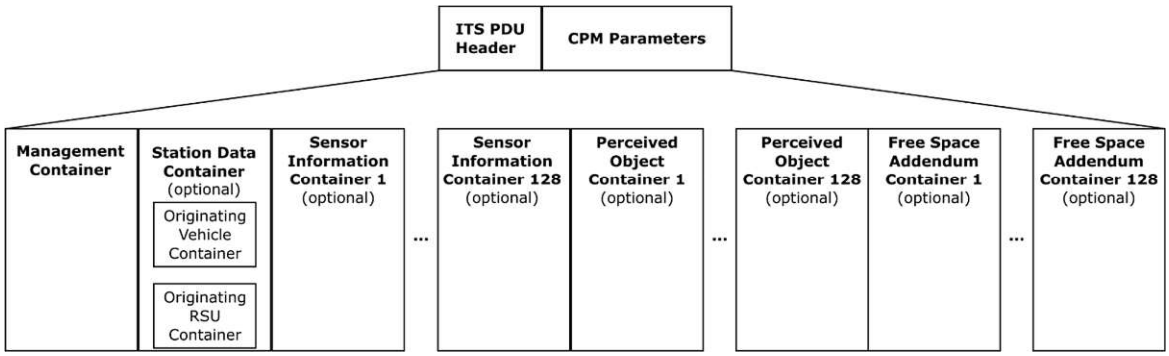

Figure 2.4: In the above figure, the basic structure of a Collective Perception Message (CPM) is shown [1].

2.3.3 Use-Cases

Applying the previously presented CPMs, several use-cases arise for which CP can be particularly useful. They are briefly described below.

Detection of Non-Connected Road Users

Road users who are not able to communicate by themselves can only be perceived by sensors of other road users. If so, the perception range is limited to the sensors' field-of-view, which is particularly critical for objects that are occluded by obstacles. With the CPS, the number of road users detected and shared by entities using the CPS can be significantly increased. In addition, as the number of entities sharing information about the same object increases, the accuracy of the estimated parameters (such as object position, speed, etc.) increases [1].

Detection of Safety-Critical Objects

In addition to road users who cannot communicate, there may be unwanted objects on or near the road that could become a potential safety hazard to road users. These items may be lost cargo, a tree stump, or debris on or adjacent to the roadway. Sharing information about these objects enables road safety applications that warn approaching entities such as vehicles of their presence. In addition, road users who are not equipped with sensors or whose sensors cannot detect these safety-critical objects can also be warned [1].

CAM Information Aggregation

Sometimes it can be useful to not only consider sensor data but to also include information from CA messages (CAM's) to generate and send out a CPM. In this work, however, the aggregation with CAM's is not considered [1].

2.3.4 CPM Dissemination Concept

Channel overload is an issue that can easily occur when sending CPM's over the V2X network. Each entity is usually generating multiple CPM's per second, sharing their sensor measurements and state information to others. Depending on the amount of sensors, this can result in very high data throughput.
Therefore, it can be useful to define CPM generation rules, aiming to balance frequent updates about detected objects and to minimize channel utilization [1].
According to [1], there are many ways of managing CPM generation. As an example, CPM Generation Frequency Management aims to manage the inclusion of the Perceived Object Containers and Sensor Information Containers.

2.4 Challenges

There are many challenges coming with the use of a CPS, as discussed in [48]. Channel overload due to huge amounts of data sent between entities such as vehicles or RSU's has already been discussed in section 2.3.4. This section discusses two other very critical issues: the impact of severe weather on data accuracy and the potential trust and safety risks associated with using the resulting erroneous CPM's. Due to the importance of this topic and the fact that it has not been adequately addressed in current CPS algorithms, as shown in chapter 3, it is the focus of this work.
At this point it has to be mentioned that there are several other issues that may

(indirectly) affect the CPS and the measured data, such as electromagnetic interference or the ambient temperature, but these are not addressed in this work.

2.4.1 Impact of Adverse Weather Conditions

Currently, one of the most critical problems in the development of AV's is the poor performance of their sensors in adverse weather conditions such as rain, snow, and fog. This generally results in false readings, which in CPS terms can mean the following: Vehicles that send false information should be considered less of a source of information. It is therefore critical to consider the performance of individual sensors under certain weather conditions [51].
In this section, the common sensor types used in AV's, mentioned in section 2.1.1, are being reviewed for various weather and lightning conditions.

Precipitation (Rain, Snow)

Precipitation is generally understood to be water in either a liquid or frozen state. The size and the distribution of the water droplets define the intensity of precipitation. The intensity, in turn, can affect sensor readings from a LiDAR, for example, since they must propagate through the precipitated medium. If the water droplet's diameter exceeds 6 mm, it will be subject of so-called Mie scattering [52]. Mie scattering can affect the propagation in two ways: firstly, due to the absorption of electromagnetic energy by the water droplets, which leads to attenuation, and secondly, due to the volume back scattering or rain clutter, which can lead to incorrect measurements [25].
LiDAR sensors in their 905 nm to 1550 nm wavebands are heavily affected by Mie scattering from rain [53]. It has been shown, however, that slight rain conditions will not have a big impact on the LiDAR's range of visibility, while more heavy rain indeed will, as shown in [54]. As mentioned in [51], there are already algorithms that can correct the image by filtering out raindrops or snowflakes with pixel-oriented evaluation. However, there are no studies yet on the accuracy that these algorithms can achieve under real adverse weather conditions.
For RADAR systems, the effect of Mie scattering is not significant at short distances, it can however decrease the maximum range of detectability, as discussed in [55, 56]. The reason is that the size of the droplets is comparable to the wavelength of the RADAR. The attenuation effect reduces the received power of the signals and the back scatter effect increases the interference at the receiver. A detailed analysis on the corresponding effects can be found in [51].
Cameras usually rely on scene brightness to determine intensity of pixels. Adverse weather conditions however can result in sharp intensity fluctuations which can decrease

the quality of images or videos. Rain or heavy snow can increase image intensities and obscure edges of the patterns of objects, which makes it hard to detect them correctly [25]. There are mainly two different technologies used to solve this problem: real-time processing and post-time processing. In [51] they are explained in more detail.
Ultrasonic sensors in general are barely affected by scattering effects which is why precipitation is not a big problem for them, as stated in [25].
According to [57], GNSS position sensors are generally not affected by local weather conditions, because of their specific signal frequency of approximately 1.575 GHz. However, they can be affected by space weather. For example, irregularities in the ionospheric layer can lead to rapid fading of signal power due to destructive interference. This effect is called ionospheric scintillation and is described in [58].

Fog

Under fog someone typically understands a visible aerosol consisting of small water droplets suspended in the air or near Earth's surface [59]. For fog to occur, the air must contain either dust or air pollution. If so, water vapor condenses around these very small solid particles, generation droplets of the size from 1 to 20 microns [60].

LiDAR sensors are very much affected by fog and back scattering occurs because their operating wavelengths are smaller than the size of the fog particles. However, the strongest effect is attenuation, which leads to very high extinction coefficient, which in turn drastically reduces the detection capability of LiDAR' sensors [61].
Fog can indirectly affect RADAR if temperature requirements are met by condensing on the radome or target in question, mimicking what is explained in the section about precipitation [25].
Cameras are severely affected by fog because their operating wavelengths in the 400 nm to 750 nm range are smaller than the size of the fog particles, which in turn leads to Mie scattering effects. As with precipitation, fog can also reduce the contrast of the image and make it difficult to detect pattern edges [62]. According to [25], the presence of so-called air-light should also be considered when examining the performance of cameras in fog. Air-light can be defined as the scattering of light by the interference of fog particles. As stated in [25], in the presence of air-light it is almost impossible to detect objects near the light source.
Even though ultrasonic sensors are not particularly affected by adverse weather conditions, they work with sound waves, so air conditions and temperature fluctuations could theoretically affect their performance. Since this type of sensor is only used for very short distances, the effects of fog and precipitation are minor [25, 51].
Because of their specific signal frequency, GNSS sensors are not affected by foggy con-

ditions [25].

Light Conditions

Having Poor or too bright light is a special and very critical problem for cameras and partially LiDAR's. It can be caused by the sun or artificial sources like air-light pollution form a skyscraper. In [63] it was shown that both a normal visual camera and a LiDAR sensor perform very poorly when trying to measure the environment near the light source, while a thermal imaging camera on the other hand was able to detect some objects where the normal camera could not. In addition, reflections from all kinds of shiny objects make it even more difficult to find the camera's exposure selection, and the camera's detection performance drops to almost zero [64]. One solution proposed in [65] is to use HDR cameras, which can mitigate lighting conditions: They can overcome sudden light changes that occur when entering a tunnel and have better color retention, which can improve their performance when driving in direct sunlight. There is also techniques existing for LiDAR measurements to separate them from sunlight, presented in [66].

2.4.2 Safety and Trust Issues

As previously discussed in this chapter, there are several challenges in implementing an AD system. Although many of these were local problems of a particular AV, the correctness of messages transmitted between vehicles and other entities becomes a critical issue when implementing a collective perception service. Collective perception makes vehicles vulnerable to attack, and dishonest vehicles can influence the decisions of other vehicles based on the data they send [22]. Regardless of whether the problems are caused by CPS or not, it can be said that all of them lead to the same result for an AV: They result in wrong decision-making, which in turn reduces safety. And more generally, it can be said that an AV that acts less secure should be less trustworthy.
To address all these issues and to eradicate misbehaving entities from the V2X network, the notion of trust is used. Trust can be understood as a degree of risk, vulnerability, or uncertainty that establishes an expectation about a entity's future behavior [67]. Trust is assessed using trust management models in which entities typically evaluate trust for other entities, with the evaluators referred to as trustors and those being evaluated as trustees [2].

Components of Trust

As already mentioned, trust mainly relies on the quality of interactions between entities, which can be divided into two components:

- **Direct Trust**: Exhibits direct observations of a trustor on a target vehicle (trustee), relying on the interactions between these two only [68].
- **Indirect Trust**: Considers the opinions of neighbors of a trustor about the trustee, taking into account past experience with the trustee and its historical behavior. It is said that direct trust is more important than indirect trust, but both must be considered when evaluating a trustee's trust [69]. Figure 2.5 compares the concepts of direct and indirect trust.

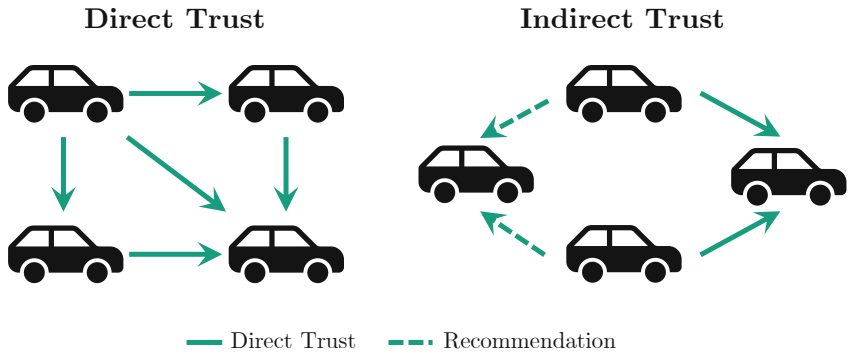

Figure 2.5: Comparison of direct and indirect trust [2].

Categories of Trust Management Models

According to [70], such trust management models can be divided into three different groups:

- **Data-Centric Models**: The correctness of the exchanged messages shared among entities are in the main focus. These models evaluate the trust of every incident, therefore, delays and data loss may be present in case of dense traffic. On the other side, data-oriented trust models do not perform well in information sparsity due to the lack of evidence.
- **Entity-Centric Models**: The credibility of vehicles and other entities is ranked based on the indirect and direct trust scores of the trustee. Therefore, sufficient data is required on the rating of a trustor and its neighbors with respect to a

trustee. It is believed that message authenticity may be a challenge, as there is no guarantee that the messages sent will not be corrupted by external sources, as described above [71].

- **Hybrid Models**: The legitimacy of both, data and entity-based trust evaluation is considered. Thus, the authenticity of the exchanged data, the recommendations of the neighbors to the trustee and their historical behavior are taken into account.

While much research is in going for all three models as described in chapter 3, in this work, the main focus lies on the entity-centric trust models. The evaluation of trust and the algorithm proposed for it form the core of this thesis and are discussed in chapter 4.

Chapter 3

Trust Management Models

This chapter discusses and compares various modern approaches for building a trust management model. According to [2], trust models can be divided into five different categories: (1) Traditional, (2) bayesian-inference-based, (3) fuzzy-logic-based, (4) machine-learning-based and (5) blockchain-based models.

3.1 Traditional

Traditional trust management models are usually widely accepted frameworks because they do not require complex data analysis or statistical inference tools for the trust evaluation. To calculate the trust value, several trust values are mainly aggregated to obtain a final value for each entity. Usually, the aggregation is done by a weighted sum with static or dynamic weights.
In [72], MARINE, a man-in-the-middle attack resistance trust model in connected vehicles is presented. MARINE corresponds to a hybrid trust model that considers a trustworthy vehicle to transmit false messages due to malfunctioning and a malicious vehicle to send a malicious message. This trust management model evaluates three different trust scores: node-centric trust, data-centric trust, infrastructure-based trust and vehicle-based trust. Node-centric trust is calculated by combining past interactions with the trustee and the opinions of its neighbors. Data-centric trust is evaluated by considering the quality of the data received and the recommendations of its neighbors by calculating direct and indirect trust values. While calculating these values, each vehicle generates a positive report that includes honest vehicles and a negative report that includes dishonest vehicles. These reports are then submitted to the RSU's, which calculate the infrastructure-based trust values and update the corresponding reports. The updated reports are then shared with all units in the network. The proposed model has been tested in a simulation of urban mobility (SUMO) and in a simulation of vehicles in the network (VEINS) for three types of attacks.

Hurl et al. [73] presents an approach for AV's to communicate perceptual observations that are mitigated by trust modeling of peers providing reports. The so-called TruPercept model makes it possible to improve an ego-vehicle's perceptual performance by fusing the received messages from other vehicles according to their reported accuracy, thus enabling collective perception. Trust is computed in three steps: In the first step, trust scores are calculated for each detection of the ego-vehicle by aggregating the trust scores from all nearby vehicles which detected the same object. Then, the trust value for each detection is adjusted by aggregating over a given time window in order to consider the time-domain. Finally, the confidence values of all detections that identified both the ego-vehicle and another vehicle are combined, leaving a single confidence value that evaluates the corresponding vehicle. Additionally, to detect malicious agents that insert false information, a plausibility checker is incorporated, that verifies LiDAR point cloud data on plausibility. Their model is evaluated using a cooperative synthetic dataset generated by a game engine in an urban environment.
Chuprov et al. [74] introduced an approach for reducing traffic management issues on crossroads by identifying and excluding vehicles sending malicious or faulty messages. They made use of three concepts, namely truth, reputation and trust, having the objective of assessing the legitimacy of the data sent. Truth defines the correctness of data being exchanged by the vehicles, reputation considers the time-evolution of truth values and trust is the weighted aggregation of truth and reputation values. After calculating a trust value for each vehicle within the network, it is compared to a set threshold, and vehicles whose value is above it are considered trustworthy. On the other hand, vehicles whose trust values are below the threshold are not trusted and are excluded from the communication network. The performance of the system is first evaluated using a customized simulator. Then, the results are validated using hardware simulations based on an autonomous vehicle model developed by the authors.

3.2 Probabilistic

Since trust is a subjective rather than a hard measured quantity, it can be well calculated using probabilistic approaches. Typically, Bayesian statistics are used in this case, utilizing a prior distribution of parameters combined with a likelihood function to generate a posterior distribution [2].
In [75], authors have proposed an anti-attack trust management scheme called AATMS that uses a TrustRank-based algorithm which was initially introduced to combat web spams. The model evaluated both global and local trust, which indicate the local and global trust relationships between vehicles. RSU's are assumed to be fully trusted. The algorithm works as follows: first, local trust is computed by applying the Bayesian infer-

ence model to past interactions. Second, a trust link graph is generated, that includes all the local trust values that vehicles gave to each other. To compute the global trust score, several additional parameters are considered: social parameters of the driver like age or driving license score, general information regarding the vehicle like its type or braking performance. All the previously mentioned parameters are then combined with the local trust as well as the past global trust scores before applying the TrustRank-based algorithm. The algorithm identifies the most trustworthy vehicles, which are also named as seed vehicles, which in turn helps to evaluate the trust of other vehicles. The model is tested using VEINS and evaluated using two introduced performance metrics: the true negative and true positive rates.

Fang el al. [76] presented a Bayesian-based trust decision scheme named BTDS that uses a Bayesian network to prevent on-off attacks. In these attacks, vehicles alternate between being honest and dishonest, sending either correct or malicious data, respectively [2]. A trust value is calculated by a weighted aggregation of direct and indirect trust values. Here, the direct trust value evaluates the direct, current, and past interactions between a trustor and a trustee and is calculated as the mathematical expectation of a Gaussian distribution. The indirect trust value corresponds to the highest direct trust value a trustee receives from all its neighboring vehicles. To identify the attack, a window is defined to examine the change in trust values. The authors performed simulations to evaluate their model using MATLAB.

Gao et al. [77] proposed a hybrid approach to evaluate trust nodes to identify and exclude malicious nodes that might intercept or discard data, which in turn disrupts the transmission process. To quantify the credibility of nodes, the concept of integrated trust is presented, which consists of direct trust and recommended trust. The direct trust is calculated by utilizing historical interaction records and Bayesian inference considering penalty factors. The recommended trust value represents the trust of third-party nodes and their reputation. Since the direct confidence value is a probability estimate with some error, a confidence value is used to describe its accuracy. If the accuracy is above a predefined threshold, the direct confidence value is considered accurate and then defined as the integrated confidence value. If the confidence value is smaller, the recommended confidence is needed, and to obtain the integrated confidence value, the direct and recommended confidence are fused. After that, a time-sliding window and a decay function are introduced to ensure timeliness and to update the integrated trust scores. Thereby, the decay function ensures that the latest information has a higher impact on the calculation of new trust values. The proposed algorithm is evaluated by means of a series of experiments and the authors propose that the method outperforms baseline methods with respect to reliability.

3.3 Fuzzy Logic-based

In general, fuzzy logic focuses on expressing the imprecision of human reasoning for decision making in an imprecise and uncertain environment. It is a generalization of standard logic, in which all statements have a truth value of one or zero. In fuzzy logic however, statements can have a partial truth value, e.g. 0.9 or 0.5. This gives the approach more opportunity to mimic real-world circumstances where statements of absolute truth or falsehood are rare [6].

Guleng et al. [78] proposed a decentralized trust management framework that used fuzzy logic to fuse the direct experience of vehicles (trustors) with recommendations of all nearby vehicles towards a target vehicle (trustee) in order to identify dishonest behavior of the corresponding target. For this purpose, both direct and indirect confidence values are calculated. To determine the direct trust value, the number of messages forwarded by the target vehicle, the number of true messages forwarded by the target vehicle and the number of intended incidents reported by the target vehicle are considered before applying fuzzy logic. The indirect trust value is calculated for vehicles that do not have a direct connection to the trust provider by applying reinforcement learning. The presented model was simulated using the network simulator NS-2.34.

In [79], authors presented a novel fuzzy-logic-based scheme for malicious node detection. It uses a unique security strategy that utilizes node behaviour during message exchange as a security metric to detect and exclude malicious nodes and their activities from the communication network. They applied fuzzy logic to generate a rank of each node named as a trust level, which describes the node's reliability in exchanging safety messages correctly. To compute trust, three criteria are mainly considered: Emulgation attack attempts, collaboration degree and RSU assessment. The model has been evaluated using MATLAB and results show that the proposed model improves the network performance, boosts network security and enhances throughput.

Hasan et al. [80] presented a hybrid fuzzy logic-based trust estimation model to adequately model malicious characteristics of a vehicle and calculate its trust. The proposed scheme uses three fuzzy set-based factors to assess the level of trust for a vehicle: (1) A Packet Drop Factor that determines a vehicle's willingness to cooperate in forwarding received packets as a function of the vehicle's speed. (2) A False Packet Injection Factor that determines whether vehicles generate false packets. (3) A Content Alteration Factor, which evaluates a vehicle's ability to modify content. After determining these factors, they are fed into a fuzzy logic-based algorithm to determine a trust level. According to the authors, their model consists of vehicles and so-called edge servers, where the latter can be understood as semi-trusted cloud servers that collect information about vehicles and calculate and update trust values in a timely manner. The

model was tested using the OMNeT++ simulation system, and the results showed that the proposed system performed better than that of Guleng et al. in terms of several metrics [78].

3.4 Machine Learning-based

Machine learning is a subfield of artificial intelligence and relies primarily on experience in the form of data to make predictions and make decisions with precision over time [81]. It can be used to classify trust levels of nodes based on measured information. There are many different types of machine learning classification techniques. The choice of the right method depends on many factors, such as access to training data, type of data, speed, accuracy, etc. However, they are often associated with a high computational cost and lack of interpretability, which makes them less suitable for many safety-related applications in practice.

Gyawali et al. [82] presented a machine learning and reputation-based misbehaviour detection system with the objective of enhancing the detection accuracy and ensuring reliability of both vehicles and messages. To do so, the model utilizes three contributing parameters known as similarity, familiarity and packet delivery ratio. The scheme works as follows: First, each vehicular message is evaluated using a machine learning model. Afterwards, the results of the evaluation are sent to a local authority, which combines them using the Dempster-Shafer theory [83, 84]. The result of this amalgamation is then used to update the reputation value of each vehicle using beta distributions. Subsequently, the reputation value is shared with other authorities and a warning is sent for revocation if the reputation value is below a certain threshold. The model was evaluated with VEINS, OMNeT++ and SUMO.

Abdelmaguid et al. [85] introduce situation awareness, a concept that uses environmental elements and events to gain a holistic view of a system at any given time. In the proposed model, situation awareness is utilized to predict trust scores of surrounding vehicles that are then applied to reevaluate the outcome of a trained machine learning model. Depending on the outcome of the situation awareness and machine learning models, a message will be classified as benign or not. The system works in two steps: First, situational awareness is measured to reflect the state of the network at a given time. This is done using a third-party ranking algorithm that uses logistic trust to rank vehicles by calculating their trust scores. Second, the holistic view of the situation awareness model is combined with the information from the machine learning model to achieve better results in detecting misbehavior attacks. The model was tested with different machine learning classification models and evaluated against the publicly available VeReMi dataset.

Huang et al. [86] present a trust management model based on machine learning and active detection technologies, to evaluate trust of vehicles and events. By using active detection technology, vehicles are able to detect the indirect trust of their neighbors, which in turn can improve the speed of filtering malicious entities. Then, a Bayesian classifier is used to assess whether a vehicle is a malicious entity based on the vehicle's state information. The proposed trust management model consists of two parts: A system initialization and vehicle trustworthiness evaluation, which calculates direct and indirect trust values. After calculating the trust values, the authors use a blockchain to store the corresponding values and certificates, which ensures the consistency of vehicle information in different domains and limits the influence of malicious vehicles or RSU's. To verify the feasibility and performance of the proposed model, it was implemented in Python.

3.5 Blockchain-based

The blockchain is a computational paradigm that emerged with the Bitcoin protocol in 2008 [87]. In general, blockchain technology deals with the distributed digital ledger of transactions. It consists of an immutable decentralized database in the form of data blocks that form chains [88]. Being a secure and decentralized computing infrastructure, it is widely recognized as a breakthrough solution to the problems of centralization of centralization, privacy and security in storing, tracking, monitoring, managing and sharing data [89].

Javaid et al. [90] propose DrivMan, a system model using blockchain and a certificate authority to provide trust management. The aim of this system is to register vehicles and, if necessary, to revoke their registration in order to promote the verification of information. To ensure the trustworthiness of the data, so-called physically unclonable functions are used. After data generation, a list of trusted registered vehicles is checked for the originating vehicle. A certificate is issued by a certification authority (e.g., an RSU) if the system successfully finds the vehicle in the trusted list and the responses of the physical unclonable functions are correct. To simulate and test the model, an Ethereum virtual machine was used in combination with a threat model. The threat model provides an attacker that can mimic a vehicle and transmit forged information to the certificate authority.

Ghovanlooy et al. [91] provides a scalable blockchain trust management system that focuses on the reliability of incoming messages on the network. For this purpose, the vehicles continuously check the validity of the received messages using a proposed Bayesian equation and specific information stored in the blockchain. After the validation process, depending on the result, the vehicle calculates a rate for each message type and

vehicle from which the message originated. To calculate the net reliability value, vehicles must upload their estimated rates to the RSU, which then applies a sharding consensus mechanism to calculate a level of trust and to generate blocks. The most recently updated blockchain is then maintained for all RSU's. The proposed model is simulated and evaluated using a python based simuator.
In [92], the authors presented a blockchain-based anonymous reputation system to develop distributed trust management while protecting vehicle privacy. The system incorporates a trust model to improve message trustworthiness based on the reputation of the sending vehicle derived from both direct historical interactions and indirect opinions of neighbors about the sending vehicle. The scheme works as follow: First, anonymous authentification has to take place, which is fundamental to trust communication and privacy protection. To do so, the system has to be initialized, certificates have to be updated and public keys have to be revoked. Second, the trust level of the broadcasted messages are to be estimated, which in turn relies on a reputation score of the sending vehicle. This reputation score is evaluated depending on the broadcasting pattern and the message priority levels.

3.6 Comparison

In the previous sections, different types of trust management models were presented. Table 3.1 compares the proposed models in terms of whether they take into account weather conditions, multiple entities such as RSU's or road users transmitting data, or whether they are able to detect misbehaving entities.
It can be clearly seen that almost all of them are able to detect misbehavior, most of them also consider RSU's in their network. However, to the best of my knowledge, none of the currently existing trust management models include the impact of context information such as weather conditions in their trust management models.

Table 3.1: Comparison of Trust Management Models

Ref.	Category	Proposed Scheme	Model Type	Evaluation Tools	Misbehaviour Detection	Weather Consideration	RSU/road user in network
[72]	Traditional	MARINE: Man-in-the-Middle Attack Resistant Trust Model in Connected Vehicles	hybrid	VEINS, SUMO	✓	–	RSU only
[73]	Traditional	TruPercept: Trust Modelling for Autonomous Vehicle Cooperative Perception from Synthetic Data	entity-centric	Python-based simulator	✓	–	✗
[74]	Traditional	Reputation and Trust Approach for Security and Safety Assurance in Intersection Management System	entity-centric	Custom software simulator	✓	–	✗
[75]	Probabilistic	AATMS: An Anti-Attack Trust Management Scheme in VANET	entity-centric	VEINS, OMNeT++, SUMO	✓	–	RSU only
[76]	Probabilistic	BTDS: Bayesian-based trust decision scheme for intelligent connected vehicles in VANETs	entity-centric	MATLAB	✓	–	✗
[77]	Probabilistic	A Hybrid Approach to Trust Node Assessment and Management for VANETs Cooperative Data Communication: Historical Interaction Perspective	hybrid	Manhattan mobile model	✓	–	✗
[78]	Fuzzy Logic	Decentralized Trust Evaluation in Vehicular Internet of Things	hybrid	NS-2.34	✓	–	✗
[79]	Fuzzy Logic	A Novel Fuzzy Logic-Based Scheme for Malicious Node Eviction in a Vehicular Ad Hoc Network	hybrid	MATLAB	✓	–	RSU only
[80]	Fuzzy Logic	A Fuzzy Logic-Based Trust Estimation in Edge-Enabled Vehicular Ad Hoc Networks	hybrid	OMNeT++	✓	–	✗

✓Adressed, ✗ Not Adressed, – Not Mentioned

Table 3.1: Cont.

Ref.	Category	Proposed Scheme	Model Type	Evaluation Tools	Misbehaviour Detection	Weather Consideration	RSU/road user in network
[82]	Machine Learning	Machine Learning and Reputation Based Misbehavior Detection in Vehicular Communication Networks	hybrid	VEINS, SUMO, OMNeT++	✓	–	✗
[85]	Machine Learning	SAMM: Situation Awareness with Machine Learning for Misbehavior Detection in VANET	hybrid	VeReMi	✓	–	✗
[86]	Machine Learning	Trust Management Model of VANETs Based on Machine Learning and Active Detection Technology	entity-centric	Python	✓	–	RSU only
[90]	Blockchain	DrivMan: Driving Trust Management and Data Sharing in VANETs with Blockchain and Smart Contracts	entity-centric	Ethereum virtual machine based simulations	✓	–	RSU only
[91]	Blockchain	SBTMS: Scalable Blockchain Trust Management System for VANET	data-centric	Python-based simulator	✓	–	RSU only
[92]	Blockchain	BARS: A Blockchain-Based Anonymous Reputation System for Trust Management in VANETs	hybrid	Custom software simulator	✗	–	RSU only

✓Adressed, ✗ Not Adressed, – Not Mentioned

Chapter 4

Proposed Model

As mentioned in chapter 3, none of the currently existing algorithms take into account the impact of contextual information on the computation of trust values. As a result, those trust management models can produce inaccurate predictions, which in turn may lead to wrong decision-making of AV's.
In this work, a novel approach is proposed to address this issue, which is explained in the following sections. The discussed framework is based on the approaches from [93], [75], [74], however, new methods to compute the respective values are defined. In the beginning of this chapter, the main components describing the trust management model are introduced. Then, the framework and its scheme is described in more detail to show the functionality.
The presented model is implemented using Python [94].

4.1 Model Architecture

4.1.1 Agent

The key component of the whole framework is called the agent. An agent in the sense of this work is known as any entity that is part of the V2X communication network making use of CPS. Here, it is assumed that the agent can be a vehicle as well as a RSU.

4.1.1.1 Properties

Specifically for this model, each agent has certain properties to provide base information. They include a class name to know whether its a vehicle or a RSU, a clear identity number, what kind of sensors it is equipped with as well as a value describing its initial reputation. The latter one will be further discussed in section 4.3.5.

4.1.1.2 Information Storage

Each agent is collecting and storing the information it has measured at discrete increments of time, also known as time frames. The following information is collected for each time frame:

- **Pose information**: Values describing the pose of the agent, which is determined by the x-y-z position as well as the orientation using pitch θ_a, roll ϕ_a and heading angle ψ_a.
- **Sensor information**: Information associated with the sensors such as their type or detection range. It also contains information about the local position of the sensors within the agent, which is required to transform the measured signal from the local sensor coordinate system to a global reference coordinate system.
- **Perceived objects**: Information regarding all the objects that have been detected by the agent. This information includes the points of the point cloud associated with the object as well as a two-dimensional bounding box computed by an classification algorithm. More information about the used classification algorithm can be found in section 4.3.2.
- **Reputation and Trust**: These values associated with each agent are computed for each frame in time, for more information see sections 4.3.5 and 4.3.4, respectively.

4.1.2 Central Server

The central server is the other key element of this algorithm. It receives all the measurement data from surrounding agents and performs computations to evaluate their performance. In this work, it is assumed that the central server has enough computation power to run the algorithm in real time.

4.2 Occupancy Grid

In this model, an occupancy grid is introduced, which lies in the idea to model roads and parts of the earth with the help of grid blocks, directly connected to each other. The dimensions of the grids and their grid spacing can vary depending on the location that should be analysed. However, in this work the grid is assumed to be two-dimensional. Each grid block is associated with a central server, that receives the information sent by the agents located in the same grid block. Through this separation into grid blocks,

the amount of data sent between agents and central server should be reduced, since only agents close to the scenery are taken into account for further computations. An example of the structure of grid blocks is shown in figure 4.1.

Figure 4.1: Example of a two-dimensional occupancy grid block modeling a road intersection [95].

Each of the grid blocks consists of cells, that can be occupied with a certain occupancy probability $P[O^c]$. This probability of occupancy is representing the main outcome of this model and should help individual agents in locating arbitrary objects that may be outside their FoV. Its value is calculated at the central server and described in more detail in section 4.3.3.

4.3 Scheme Overview

The presented algorithm scheme can mainly be divide into two parts, the agent-side computation and the server-side computation.
The former is shown in 1 and mainly incorporates two steps, which are 1) measuring the environment and 2) performing the object detection. The two steps will be described in more detail in sections 4.3.1 and 4.3.2, respectively. After the agent-side computation is done, the agent sends a CPM to the central server, where further computation is done. The server-side computation is shown in 2. It first receives all the CPM's of the agents located in the grid block. Then, it maps all the detection information to the grid block and computes occupancy probabilities for each of its cells considering current

reputation levels of the agents. Afterwards, values for trust and an update of the reputation values are computed. While reputation and trust will temporary be stored on the central server, values for the occupancy probabilities will be sent back to the agents and deleted afterwards.
To be related to standard trust management model notation, the central server can be seen as the trustor, that is assessing all the agents located inside the grid block, that play the roles of trustees. However, this notation will not be used in this work.

Algorithm 1: Agent-Side Computation

Input: -

Output:

CPM_a collective perception message containing all data of agent a for current time frame

1 $CPM_a := \{\}$ // initialize CPM
2 $CPM_a \leftarrow$ **add**$(properties)$ // add general agent info
3 $data_a =$ **sensor_reading**() // section 4.3.1
4 $objects =$ **object_detection**$(data_a)$ // section 4.3.2
5 $CPM_a \leftarrow$ **add**$(objects)$ // save results
6 **send** CPM_a // send to central server

Algorithm 2: Server-Side Computation

Input:

$\boldsymbol{CPM}$ list of collective perception messages of all agents located in grid block at current time frame

Output:

$P_a[O^c]$ agent occupancy probability for any cell c
$P[O^c]$ collective occupancy probability for any cell c
$\mathcal{T}_a$ trust score for every agent a
$\mathcal{R}_{a,new}$ updated reputation score for every agent a

/* compute occupancy probabilities */
1 $P_a[O^c], P[O^c] =$ **occupancy_grid_inclusion**$(\boldsymbol{CPM})$ // section 4.3.3
/* compute agent-based values */
2 $\mathcal{T}_a =$ **compute_trust**$(P_a[O^c], \boldsymbol{CPM})$ // section 4.3.4
3 $\mathcal{R}_{a,new} =$ **update_reputation**$(\mathcal{T}_a, \boldsymbol{CPM})$ // section 4.3.5
4 **send** $P[O^c]$ // send to agents

4.3.1 Sensor Reading

At first, the agent is measuring the environment with its available sensors. In this work, it is assumed that each agent is equipped with two types of sensors: 1) a RGB camera and 2) a depth camera. The first one acts as a regular camera and captures images from the scene, which are subsequently used from the detection algorithm to detect objects. The latter one is a sensor specifically designed for the CARLA Simulator [96] which provides raw data of the scene codifying distance for each pixel of an image which allows creating a depth map. More information about the exact usage of the sensors is explained in section 4.3.2. The main specifications can be seen in table 4.1. Therein, vFoV and hFoV mean the vertical and horitontal FoVs of the cameras.

Sensor	Setting	Value	Unit
RGB Camera	resolution	1280x720	pixel
	hFoV	90	degree
	vFoV	≈60	degree
Depth Camera	resolution	1280x720	pixel
	hFoV	90	degree
	vFoV	≈60	degree

Table 4.1: Specifications of the used camera sensors.

After measuring the environment, the data of both sensors is stored temporary and will be manipulated in the following step.

4.3.2 Object Detection

In this step, the data form the depth camera and the RGB camera will be used to detect objects and create two-dimensional bounding boxes. The principle scheme of the object detection algorithm can be seen in 3.
Hereby, the captured image *img* will first be fed into the You Only Look Once (YOLO) classification algorithm, which is briefly described in section 4.3.2.1. This algorithm outputs the *class* labels of the detected objects, a class-specific confidence score $C^o_{d,a}$ as well as a two-dimensional bounding box *bbox* lying on the image plane.
After detction, the bounding box is then fused with the depth information of the depth camera to create a three-dimensional bounding box $3D\_bbox$ of the object. This is done by the projection algorithm as described in section 4.3.2.2. It has to be noted that only the two-dimensional horizontal representation of the $3D\_bbox$ named h_bbox is used for further computation to simplify the problem.

Algorithm 3: Object Detection Algorithm

Input:

$data_a$ current measurement data of agent a

Output:

$objects_a$ detected objects of agent a

```
1 depth_img, img ← get(data_a)                          // get variable from data
2 bbox, class, C^o_{d,a} = YOLO_v7(img)                 // section 4.3.2.1
3 3D_bbox = projection_algorithm(bbox, depth_img)       // section 4.3.2.2
4 h_bbox ← 3D_bbox                                      // get horizontal bounding box
5 objects_a := {}                                       // initialize
6 objects_a ← add(h_bbox, C^o_{d,a}, class)             // save pre-processed data
7 return objects_a                                      // return data
```

4.3.2.1 YOLO Classification

In this work, detections were made using the YOLO v7 classification algorithm [97], which detects objects and computes class-specific confidence scores $C^o_{d,a}$ for each object using a single convolutional neural network. The confidence scores are computed by considering the following formula:

$$C^o_{d,a} = P_a[class|o]\, P_a[o]\, IOU^{truth}_{pred,a} \tag{4.1}$$

where $P_a[class|o]$ is a conditional class probability for the detected object o being a specific class type and $P_a[o]$ is the probability of agent a detecting the object correctly. Further, $IOU^{truth}_{pred,a}$ is the Intersection-Over-Union describing how well the prediction of agent a fits the ground truth bounding box. The scheme of the YOLO algorithm can be seen in figure 4.2.

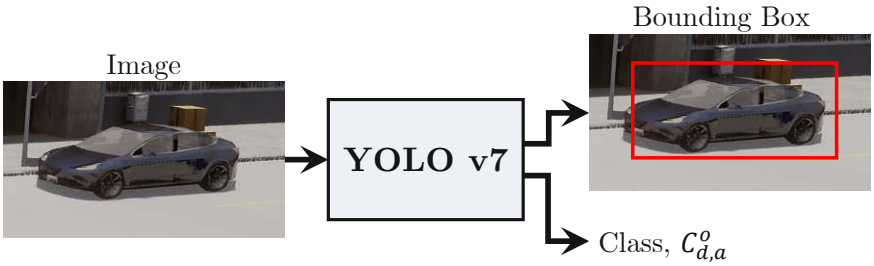

Figure 4.2: Principal scheme of the YOLO detection algorithm.

After detections were made, the algorithm outputs a two-dimensional bounding box

bbox for each detected object, the *class* label as well as the class-specific confidence score $C^o_{d,a}$.

4.3.2.2 Projection Algorithm

The two-dimensional bounding box created by the YOLO classification algorithm subsequently is used to create a three-dimensional bounding box of the object. This is done by fusing the information of the depth camera with the bounding box. Since depth information is assumed to be available for every pixel of the RGB image, the pixels lying inside the two-dimensional bounding box are simply filtered and processed with a clustering algorithm to remove ones lying significantly far away from the object cluster. The principal scheme of the algorithm is shown in figure 4.3.

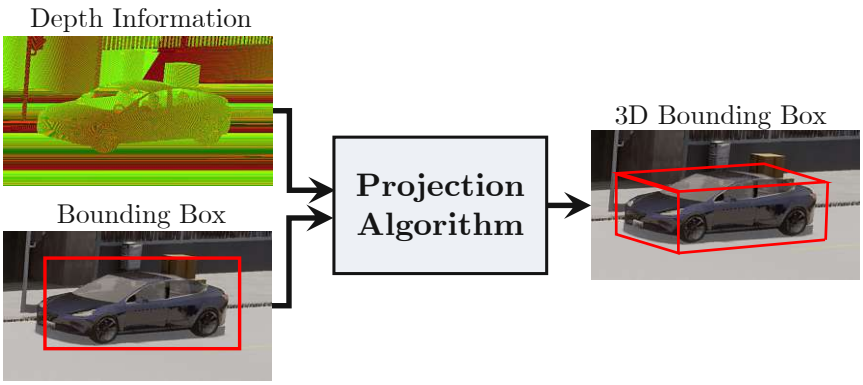

Figure 4.3: Overview of the projection algorithm fusing the depth information with the two-dimensional bounding box to generate a three-dimensional bounding box.

4.3.3 Occupancy Grid Inclusion

This step presents the inclusion of the occupancy grid into the model. It uses data from the CPM's sent from the agents to compute occupancy probabilities $P_a[O^c]$ given information of one agent as well as the collective occupancy probabilities $P[O^c]$ for each cell and current time frame. The procedure to do so is shown in algorithm 4.

4.3.3.1 Get Weather Information

First, weather has to be loaded, since the model is considering weather context for computations. As already mentioned in the first chapter, there are various types of weather

Algorithm 4: Occupancy Grid Inclusion

Input:

CPM list of collective perception messages of all agents located in grid block at current time frame

Output:

$P_a[O^c]$ agent occupancy probability for any cell c

$P[O^c]$ collective occupancy probability for any cell c

$data_w =$ **get_weather_information**() // get current weather data

for each *agent a* **do**

  $bbox,\ C^o_{d,a},\ label \leftarrow$ **get**(CPM_a)

  $C_{m,a} =$ **compute_measurement_confidence**$(data_w)$ // section 4.3.3.2

  $\mu_a^{c,o} =$ **compute_membership_value**$(bbox)$ // section 4.3.3.3

  $P_a[O^c] = \mu_a^{c,o}\, C^o_{d,a}\, C_{m,a}$ // section 4.3.3.4

  $\mathcal{R}_a \leftarrow$ **get**$(\boldsymbol{CPM})$ // get current reputations of agents

  $w_a = \left(\sum_{i \in \mathbb{N}^c} \mathcal{R}_i\right)^{-1} \mathcal{R}_a$ // agent opinion weight

$P[O^c] = \sum_{i \in \mathbb{N}^c} w_i P_i[O^c]$ // section 4.3.3.4

return $P_a[O^c],\ P[O^c]$ // return the probabilities for each cell

conditions that can negatively affect the performance of certain sensors. To reduce complexity, however, this work only deals with the following weather information: the sun altitude and azimuth angles α_s and γ_s, respectively. The angles are defined according to figure 4.4, where the drawn coordinate system represents the global system for the respective grid block.

One reason for choosing the effect of strong light only lies in the fact that it mainly affects the camera sensor. Since weather effects on other sensors cannot directly be observed in the simulator I used to test the model, only impacts on the camera are considered. The model can however easily be expanded if weather effects on other sensors can be considered as well. The other reason for choosing these two parameters is that they are physically comprehensible and follow a clear sense, thus making it easier to understand their impact.

4.3.3.2 Compute Measurement Confidence

In the next step, the so-called measurement confidence level $C_{m,a}$ is calculated, a value that assesses a very critical issue described in the first chapter representing the core of this work: how likely the detection is to be reasonable or not, given certain weather context.

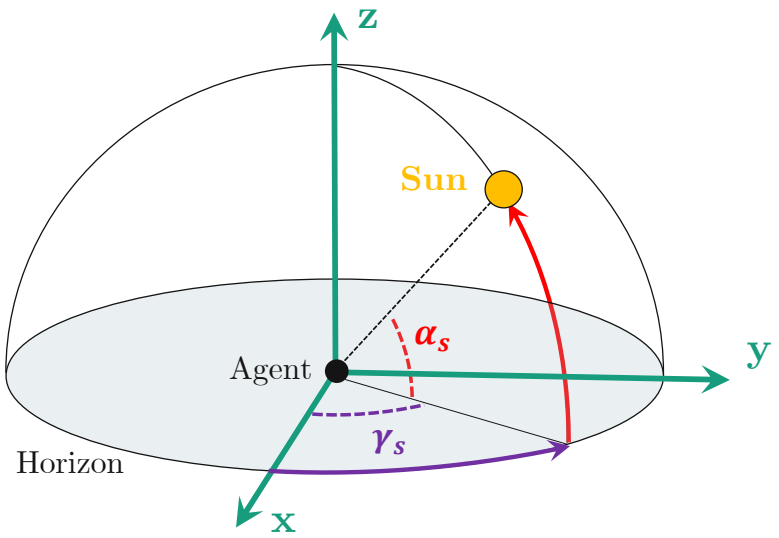

Figure 4.4: Azimuth and altitude angles γ_s and α_s for the sun with respect to the global x-y-z coordinate system.

This metric, having values between 0 and 1, should give information about the fact, that specific sensor types are very sensitive to certain weather conditions and therefore cannot be able to measure correctly. In order to calculate the measurement confidence $C_{m,a}$ of agent a being able to measure any object correctly, Fuzzy Logic is used.
In many real world scenarios, where one cannot determine whether a state is true or false, Fuzzy Logic scores with its valuable flexibility of reasoning, allowing to consider the inaccuracies or uncertainties of of any situation. In general, a Fuzzy Logic System $\mathcal{FLS}$ consists of four parts:

- **Rule Base:** Contains a collection of if-else based rules provided by experts to manage the decision-making system determining whether certain states affect a sensor or not. For each sensor type, a different rule base is created. The full set of the linguistic rules used for this model can be found in table 4.2.
- **Fuzzifier:** It converts so called crisp inputs into fuzzy values by using membership functions. The crisp inputs are basically the exact inputs we can measure (or calculate).
- **Fuzzy Inference Engine:** It represents the key unit of any fuzzy logic system and maps given fuzzy input values to fuzzy output values using if-else rules from the rule base. In this work a Mamdani inference engine is used, that utilizes linguistic rules.

- **Defuzzifier:** It converts the fuzzy output values back to crisp outputs by using membership functions. In this model, the center-of-gravity method was chosen for computing the crisp output. For further explanations the reader is referred to [98].

In order to classify our inputs and outputs, so called membership functions need to be defined. They determine, how each point in our input or output space is mapped to a membership value between zero and one. In this work, two different crisp inputs are used for the $\mathcal{FLS}$ to determine the measurement confidence: the relative pitch $\tilde{\theta}_a(\alpha_s)$ and the relative heading angles $\tilde{\psi}_a(\gamma_s)$ of agent a with respect to the sun altitude and azimuth angles α_s and γ_s. They both determine how direct the sun is shining onto the front of agent a, given the assumption that the only camera used for detecting objects is mounted there. The relative angles are calculated as follows:

$$\begin{aligned} \tilde{\theta}_a(\alpha_s) &= \min\left\{|\alpha_s - \theta_a|, \frac{\text{vFoV}}{2}\right\} \\ \tilde{\psi}_a(\gamma_s) &= \min\left\{|\gamma_s - \psi_a|, \frac{\text{hFoV}}{2}\right\} \end{aligned} \tag{4.2}$$

To give a better idea of how they are calculated, figure 4.5 shows the corresponding angles of the camera (equal to the orientation of the agent body frame) and the sun with respect to the global coordinate system of the grid block.

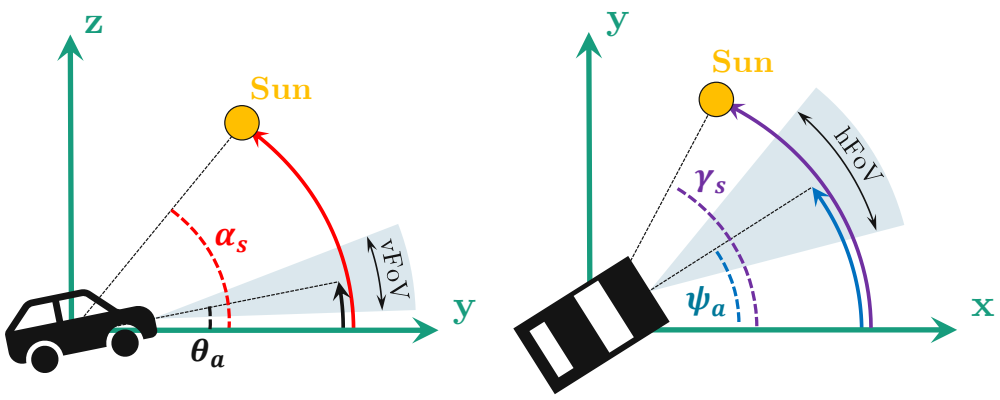

Figure 4.5: Comparison of the agent and the sun orientation.

Computing the angles as in equation 4.2 ensures that $\tilde{\psi}_a(\gamma_s)$ and $\tilde{\theta}_a(\alpha_s)$ stay within the respective FoVs. If the relative angles get bigger than half of the FoVs, then it the sun is assumed to not affect the camera measurements anymore. That is why looking at bigger angles than the corresponding vertical and horizontal FoVs is not necessary. The membership functions for the inputs and the output can be seen in figure 4.6.

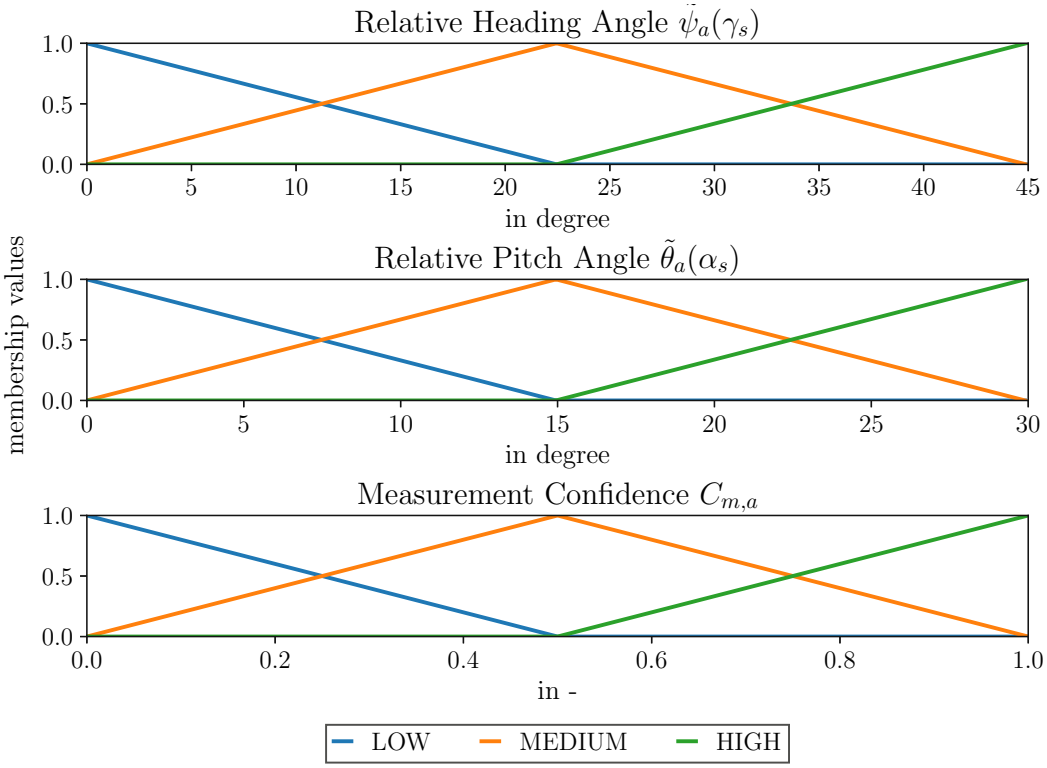

Figure 4.6: Membership functions used to describe the crisp inputs and the crisp output.

Finally, the relationship between the measurement confidence and the weather information can be written as follows:

$$C_{m,a} = \mathcal{FLS}\left(\tilde{\psi}_a(\gamma_s),\, \tilde{\theta}_a(\alpha_s)\right) \tag{4.3}$$

whereas $\mathcal{FLS}$ is the fuzzy logic system using the rule base shown in table 4.2.

4.3.3.3 Compute Membership Value

To compute the membership value of each cell being part of the object, it is generally assumed that the value equals one, if the cell lies within the two-dimensional bounding box representing the object o. On the borders of the bounding box, the membership value is assumed to decrease with a Gaussian. Let $\mathbb{C}^o$ be the set of cells lying inside the bounding box of object o, then the membership value for each cell is calculated as

$$\mu_a^{c,o} = \begin{cases} 1 & \forall c \in \mathbb{C}^o \\ e^{-\frac{r^2}{2\sigma^2}} & \text{otherwise} \end{cases} \tag{4.4}$$

If-Then Rules	$\tilde{\psi}_a(\gamma_s)$	$\tilde{\theta}_a(\alpha_s)$	$C_{m,a}$
Rule 1	LOW	LOW	LOW
Rule 2	LOW	MEDIUM	LOW
Rule 3	LOW	HIGH	MEDIUM
Rule 4	MEDIUM	LOW	LOW
Rule 5	MEDIUM	MEDIUM	LOW
Rule 6	MEDIUM	HIGH	HIGH
Rule 7	HIGH	LOW	HIGH
Rule 8	HIGH	MEDIUM	HIGH
Rule 9	HIGH	HIGH	HIGH

Table 4.2: Rule Base used to determine the fuzzy output $C_{m,a}$, by the fuzzy inputs $\tilde{\psi}_a(\gamma_s)$ and $\tilde{\theta}_a(\alpha_s)$

In the above equation, r is the distance between the cell and the closest border of the bounding box and σ is the standard deviation controlling the width of the "bell". The standard deviation σ is assumed to have a value of 0.4, and the maximum padding r_{max} around the bounding-box is defined to be the 95% confidence interval bound, which is according to the empirical rule:

$$r_{max} = 2\sigma \tag{4.5}$$

An example showing the membership values for an object can be seen in figure 4.7.

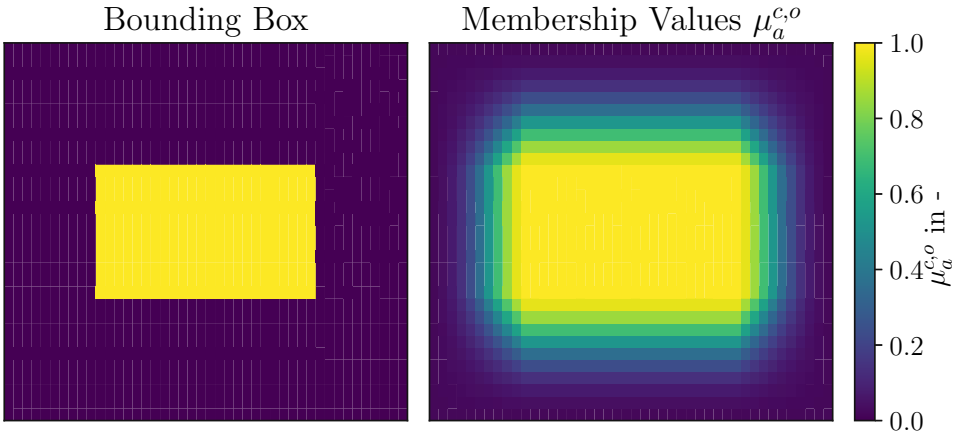

Figure 4.7: The left figure shows the actual two-dimensional bounding box and the right figure the corresponding membership values of cells being part of the object o.

4.3.3.4 Compute Occupancy Probability

The final step is computing the collective occupancy probability $P[O^c]$ for each cell c. This is done by aggregating the information of all agent occupancy probabilities $P_a[O^c]$, which can be seen as opinions about the cell states given from specific agents. This combination of opinions is done using the linear pooling method [99]. Its pooling function looks like the following:

$$P[O^c] = \sum_{a \in \mathbb{N}^c} w_a P_a[O^c] \tag{4.6}$$

Hereby, $P_a[O^c]$ are the agent occupancy probabilities for cell c being occupied given from the opinion of agent a alone. It is computed as:

$$P_a[O^c] = \mu_a^{c,o} \, C_{d,a}^{o} \, C_{m,a} \tag{4.7}$$

Further, w_i are fixed non-negative weights with sum-total of 1, that should reflect the agents' competence. This allows more competent agents to have a greater influence on the collective occupancy probability. The competence value is assumed to be equal to the agent's current reputation score $\mathcal{R}_a$, further described in section 4.3.5. The weights are calculated as:

$$w_a = \frac{1}{\sum_{i \in \mathbb{N}^c} \mathcal{R}_i} \mathcal{R}_a \tag{4.8}$$

where $\mathbb{N}^c$ is the set of agents that are measuring the cell c. In the case of only agent a measuring the cell c or in the case of the other agents to have reputation values equal to zero, w_a in equation 4.8 is equal to 1 and the collective occupancy probability becomes:

$$P[O^c] = P_a[O^c] \tag{4.9}$$

4.3.4 Compute Trust

In this section, the procedure of calculating trust for each agent is shown. In general, trust should be a measure describing how trustworthy the measurements of a specific agent are. The Trust value $\mathcal{T}_a$ for every agent $a \in \mathbb{A}$ and every time frame is computed in two steps: (1) by computing trust values $\mathcal{T}_a^c$ for each grid cell $c \in \mathbb{C}_a$ measured by agent a and (2) through fusing them by performing a weighted averaging aggregation.

Trust for each gird cell c is computed by comparing the assessments of all the agents measuring the cell. To do so, every measurement of the agents first has to be translated

into a binary value by assessing each grid cell c with a state $x_a^c \in \mathbb{X} = \{O, F\}$. Hereby, O means the cell is occupied and F means the cell is free. To determine this state x_a^c, the following cases have to be distinguished:

$$x_a^c = \begin{cases} O & P_a[O^c] > \kappa \\ F & P_a[O^c] \leq \kappa \end{cases} \tag{4.10}$$

In equation 4.10, $P_a[O^c]$ is the previously computed agent occupancy probability for cell c given that information of agent a is correct. It can be observed that the cell should be only marked as occupied, if the probability is greater than a predefined threshold κ, which is assumed to be 0.5. After each agent made its assessment for each cell c, Trust $\mathcal{T}_a^c$ for the respective agent a can be computed in the following way:
Assuming that $\mathbb{N}^c$ is the subset of agents measuring the same cell c and $\mathbb{K}_a^c \subseteq \mathbb{N}^c$ is a subset of agents assessing the cell the same way as agent a. To compute the trust value $\mathcal{T}_a^c$ of agent a, one has to consider two cases as presented in equation 4.11: If more than one agent is measuring the cell c, trust is computed according to the first case. If only agent a is measuring the cell, however, then trust for this cell should be equal a default trust value, in this case 0.5.

$$\mathcal{T}_a^c = \begin{cases} \dfrac{1}{\sum\limits_{i \in \mathbb{N}^c} \mathcal{R}_i} \sum\limits_{j \in \mathbb{K}_a^c} \mathcal{R}_j & |\mathbb{N}^c| > 1 \\ 0.5 & |\mathbb{N}^c| = 1 \end{cases} \tag{4.11}$$

Above, $\mathcal{R}_i, \mathcal{R}_j$ are values for the current reputation of an agent, described in section 4.3.5. After computing the trust values for every cell, the trust value $\mathcal{T}_a$ for agent a can be computed according to

$$\mathcal{T}_a = \frac{1}{\sum\limits_{c \in \mathbb{C}_a} w^{c,o}} \sum_{c \in \mathbb{C}_a} \mathcal{T}_a^c \, w^{c,o}. \tag{4.12}$$

In 4.12, $w^{c,o}$ are object-depending weights that consider the importance of measuring specific object classes correctly and $\mathbb{C}_a$ is the subset of cells measured by agent a. Assuming that only vehicles exist in the scenery the weights are:

$$w^{c,o} = 1 \tag{4.13}$$

For the case that agent a is detecting no object, a trust score $\mathcal{T}_a$ of 0.5 is assumed.

4.3.5 Update Reputation

Reputation is introduced to provide information regarding how well agents have performed over time. It assesses the trust values of an agent and should increase in value, if the agent performed well throughout a predefined sliding window. On the other side, it should decrease if the agent is not making correct measurements and thus not matching with the assessments of the other agents. Reputation in the context of this work can be seen as a value that requires time to evolve, it cannot change rapidly but more slowly and with inertia.
The first time agent a is sending and receiving data from the central server, it has an initial value for the reputation $\mathcal{R}_{a,0}$, that depends on how likely it is that the agent will perform well in the V2X network based on its technical features. It is assumed that agents with better autonomous driving technology or higher likelihood to measure correct have a higher initial reputation. In this work, it is assumed that a RSU is measuring worse than a vehicle and the initial reputation values are chosen according to 4.14.

$$\mathcal{R}_{a,0} = \begin{cases} 0.7 & a = \text{vehicle} \\ 0.5 & a = \text{RSU} \end{cases} \tag{4.14}$$

To model reputation in this work, a logistic growth function with variable growth rates is used, as described in the following: Logistic growth is generally described by the standard form of the logistic differential equation [100]:

$$\frac{df(t)}{dt} = kf(t)\left(1 - \frac{f(t)}{a}\right) \tag{4.15}$$

where k is the growth rate, and a is the carrying capacity, which corresponds to the maximum value the function $f(t)$ can reach. In this work, k is assumed to be variable and is determined by the current value of Trust $\mathcal{T}_a$ of the agent. It is computed as the output of a shifted sigmoid function as

$$k(\mathcal{T}_a) = c_1\left(\frac{1}{1 + c_2\, e^{-\mathcal{T}_a c_3}} - c_4\right) \tag{4.16}$$

In equation 4.16, $c_1 - c_4$ are constants used to shift the sigmoid function from its original position and scale it. In this case, these constants are tuned to let the function intersect the ordinate at a value 0.55, in order for the growth rate to be bigger than 0 only if trust values are bigger than 0.55. The resulting sigmoid function can be viewed in figure 4.8.

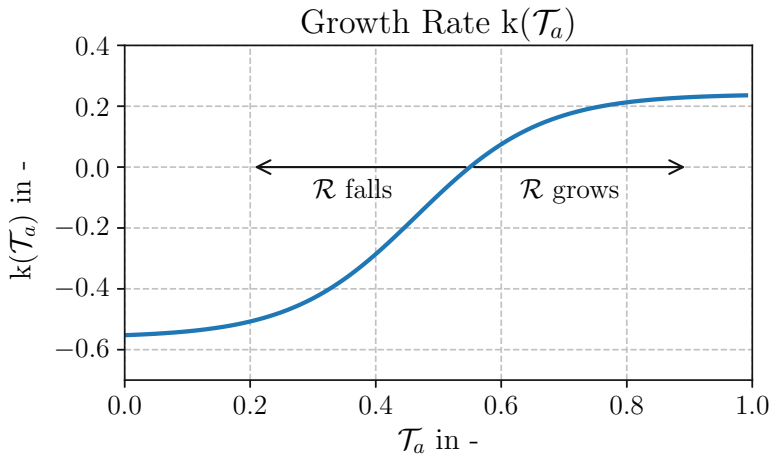

Figure 4.8: The above figure shows the function used to compute the growth rate.

A important detail about the sigmoid function defined through 4.16 is that its maximum value is smaller than the absolute minimum value, which results in reputation to decrease faster than to increase. This is a desired property of reputation and should help releasing opinion power from agents that are performing poorly as fast as possible. Finally, discretizing equation 4.15, replacing f with $\mathcal{R}_a$ and inserting 1 for a delivers the desired update function for reputation as shown in equation 4.17. It has to be noted, that for $\mathcal{R}_{a,new}$ not to remain at zero for all times, it has to be set equal to a very small value $\epsilon > 0$ in case $\mathcal{R}_{a,old}$ becomes smaller than ϵ.

$$\mathcal{R}_{a,new} = \begin{cases} \mathcal{R}_{a,old} + k(\mathcal{T}_a)\,\mathcal{R}_{a,old}\,(1 - \mathcal{R}_{a,old}) & \mathcal{R}_{a,old} \geq \epsilon \\ \epsilon & \mathcal{R}_{a,old} < \epsilon \end{cases} \tag{4.17}$$

Chapter 5

Quantitative Experiments

In this chapter, the simulation experiments conducted in this thesis will be described in detail. First, the settings used for the corresponding simulation scenarios will be elaborated and the evaluation metrics used for testing the model performance will be introduced. Afterwards, the simulation results for each scenario will be shown and compared to the others.

5.1 Simulation Design

5.1.1 Simulation Platform

As already mentioned, the simulation platform used for testing and evaluating the proposed trust management model is CARLA [96]. It is a open-source simulator running on the Unreal Engine, that allows simulating real-world scenario analysis. The main reason it has been chosen is the simulator's is not only its user-friendliness but also the flexibility in deciding which sensors should be attached. Last but not least, the model allows the consideration of weather context information, which makes it a perfect fit for analyzing the model. However, the model can be applied to any kind of data, making it theoretically usable for real-world applications.

5.1.2 Simulation Scenario

The scenario presented in this work to test the proposed model is shown in figure 5.1. It is build as follows: The environment is a T-intersection and two agents, a vehicle (agent 1) as well as a RSU (agent 2) are measuring the scenery. Another vehicle, from now on referred to as Non-Player-Character (NPC), is approaching the intersection. The NPC is not equipped with any sensors and cannot communicate with the agents, thus not being part of the V2X network. Agent 1 is not able to measure the NPC at

first, since its FoV is restricted by a building, only agent 2 has the NPC in its FoV and can eventually detect it.

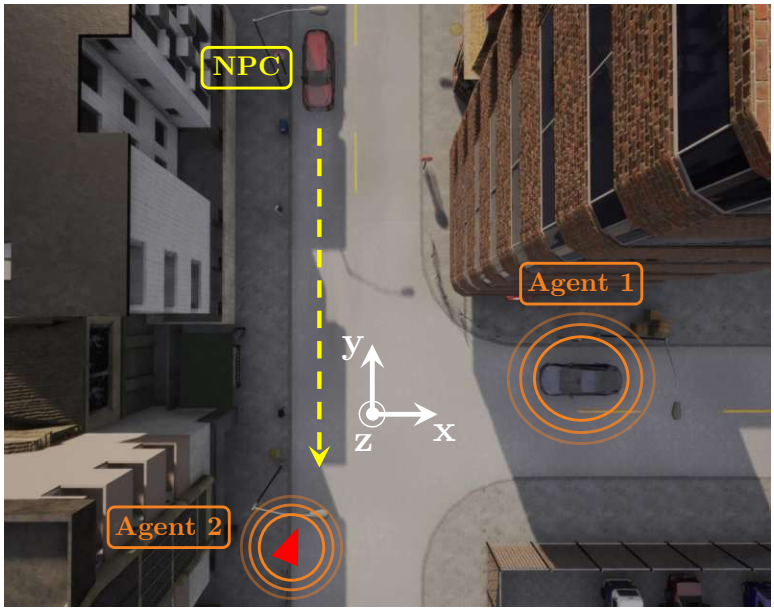

Figure 5.1: Simulation scenario proposed in this work. It consists of two agents and one NPC object, that is approaching a T-intersection.

The standard settings used in this scenario are listed in table 5.1.

Parameter	Value	Description
number of agents	2	One RSU and one vehicle
number of objects	1	The approaching vehicle
simulation time	3s	Total simulation time
frames	60	Number of time frames considered
grid size	30m x 60m	Dimension of the two-dimensional grid
grid spacing	0.2m x 0.2m	For the grid block

Table 5.1: Standard simulation parameters which are used throughout the simulation analysis.

In this scenario, none of the agents are moving, only the NPC does. The agents' only purpose is to measure their environment and detect objects. Since the only weather influence considered in this work is the impact of strong light, the exact orientation of the cameras is of high importance. The orientations of agent 1 and 2 sensors with respect to the global coordinate system are shown in table 5.2.

Agent	Property	Value	Unit
Agent 1	heading ψ_1	180	degree
	pitch θ_1	0	degree
	roll ϕ_1	0	degree
Agent 2	heading ψ_2	75	degree
	pitch θ_2	-10	degree
	roll ϕ_2	0	degree

Table 5.2: Orientation angles of agents 1 and 2 with respect to the global coordinate system shown in figure 4.5.

To demonstrate that the proposed model is considering weather context and as a result does make more accurate predictions, three scenario cases are considered. The corresponding cases are the following:

- **Case 1**: No weather context is considered, the sun is at high altitude, thus not affecting the RGB cameras.
- **Case 2**: No weather context is considered when computing the collective occupancy probabilities for all cells inside the grid. The sun is at a position such as to affect the measurements of agent 2 only.
- **Case 3**: Weather context information is considered this time. The sun is again at a position to only affect the measurements of agent 2.

By differentiating between the above cases and by comparing their results, one should get a better idea of how well the model works and why considering weather context is important for trust management models. In section 5.2, the results of the respective cases are shown in detail.

5.1.3 Evaluation Metrics

The evaluation metrics defined to measure the performance between individual agents' assements and the assessment of CP are defined in this section.
As already mentioned in section 4.3.4, one can determine the state of each cell by comparing the agent occupancy probabilities $P_a[O^c]$ as well as the collective occupancy probabilities $P[O^c]$ with a threshold κ. If the corresponding values are higher or not determines whether the cell is assessed to be occupied O or free F.
To go further with computation, some terms first have to defined, that are comparing agent and CP assessments with the ground truth:

- *True Positives (TP)*: cells that both, ground truth and agents/CP assessed as occupied
- *False Positives (FP)*: cells that agents/CP assessed as occupied, but ground truth as free
- *False Negatives (FN)*: cells that agents/CP assessed as free, but ground truth as occupied

With the above terms, the following performance metrics are defined in order to evaluate the model and compare the accuracy of agents with CP.

Precision

Precision, also known as the positive predicted value, is a metric that depicts the ability of the agent or CP to correctly predict an occupied cell. It is defined as:

$$P_{rec} = \frac{n_{TP}}{n_{TP} + n_{FP}} \tag{5.1}$$

whereas n_{TP}, n_{FP} are the numbers of cells assessed as true positive and false positive, respectively.

Recall

This metric also referred to as Sensitivity, describes how well the agent/CP correctly predicts a free cell. It is calculated by using the following equation:

$$R_{ec} = \frac{n_{TP}}{n_{TP} + n_{FN}} \tag{5.2}$$

To make a comparison between the agents, CP and the different scenario cases more concise, mean values for precision and recall are first calculated according to:

$$\bar{P}_{rec} = \frac{1}{T} \sum P_{rec} \qquad \bar{R}_{ec} = \frac{1}{T} \sum R_{ec} \tag{5.3}$$

where T is the total number of time frames considered for the simulation. Using the mean values, the F_2-Score is calculated for each agent and CP separately as described below:

F_2-Score

The more general version of the F_β-Score is a metric that combines both recall and precision to a single value. It is calculated as follows:

$$F_\beta = (1 + \beta^2) \frac{\bar{P}_{rec}\bar{R}_{ec}}{\beta^2 \bar{P}_{rec} + \bar{R}_{ec}} \tag{5.4}$$

where β is a weight that is penalizing either precision or recall more. Since in this work the main goal is to correctly measure obscured objects in order to extend the FoV, recall plays a more important role. Therefore, a β value of 2 is chosen. This results in the following equation:

$$F_2 = 5 \frac{\bar{P}_{rec}\bar{R}_{ec}}{4\bar{P}_{rec} + \bar{R}_{ec}} \tag{5.5}$$

5.2 Model Performance Evaluation

In this section, the model is evaluated by presenting the results of the three cases presented. To highlight the differences in performance between the cases, two specific time frames are considered, as described below:

- **Initial frame t_0**: This time frame marks the initial state of the scenario. Here, only agent 2 has the possibility to recognize the NPC. Agent 1's visibility is restricted by a building, which is why he cannot spot the approaching NPC.
- **Time frame t_1**: In this time frame, agent 1 can recognize the NPC in scenario case 1 for the first time. After this time frame, both agents recognize the NPC and are rated accordingly.
- **Time frame t_2**: This time frame should show the algorithm about two seconds after both agents have detected the NPC for the first time simultaneously. It should give an impression of how the corresponding ratings develop after some time.

5.2.1 Case 1: No Context Information and No Sun Impact

In the first case, no information about the weather context is taken into account when evaluating the agents. In particular, this means that the measurement confidence scores $C_{m,a}$ for each agent a do not vary, but always remains equal to 1. It is also assumed that

the sun does not affect the sensor measurements, which means that the RGB images are not distorted by sunlight. The sun orientation angles for this case are

$$\alpha_s = 90° \qquad \gamma_s = 87° \tag{5.6}$$

with respect to the global coordinate system described in chapter 4. Figure 5.2 shows the images taken by agents 1 and 2 with their RGB cameras for time frame t_2. The two-dimensional bounding box generated by the YOLO v7 algorithm is highlighted in red. It is evident that neither camera is affected by the sunlight and both detect the NPC, while Agent 2 also detects Agent 1.

(a) **(b)**

Figure 5.2: RGB images captured by agent 1 (a) and agent 2 (b) in time frame t_2 for scenario case 1. The red boxes mark detections from YOLO v7.

Then, the occupancy probabilities of agent 1 and agent 2 are compared with the collective occupancy probabilities, as shown in figure 5.3. Here, the results from three different time frames t_0, t_1 and t_2 are presented.
In figure 5.3 it can be seen that agent 1 cannot detect the NPC in time window t_0 because it is still hidden by the building. Agent 2, on the other hand, can detect the NPC from the beginning, which causes the collective occupancy probability to take values equal to the agent occupancy probability. In this case, we can say that agent 1 benefits from agent 2's discoveries and could use the grid of collective occupancy probabilities as an advantage in its decision-making process. In time frame t_1, agent 1 may discover the NPC for the first time, resulting in occupancy probabilities greater than zero. The probabilities are fused using the linear pooling method, yielding the collective occupancy grid. Time frame t_2 shows the detections approximately two seconds after time frame t_1. It indicates that agent 1 now detects the NPC with greater certainty because it is more visible to its sensors. This leads to higher values for occupancy probability.

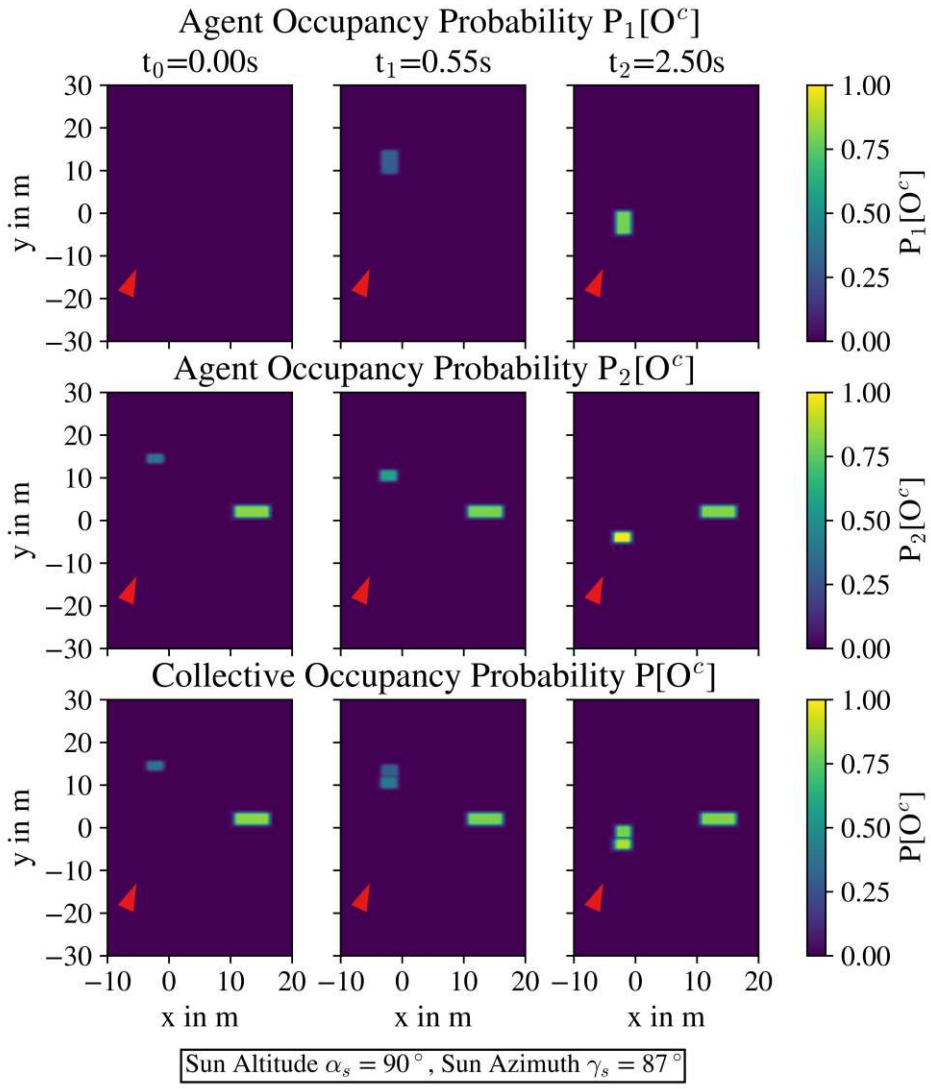

Figure 5.3: Comparison of agent occupancy probabilities and collective occupancy probabilities for all grid cells, the three time frames t_0, t_1, t_2 and scenario case 1. Hereby, the RSU is marked as a red triangle.

The trust and reputation values of agents 1 and 2 are now shown in 5.4. Agent 1 detects nothing in the first time window t_0, resulting in a trust value of 0.5. Agent 2 is the only one who detects the NPC, so the trust values of the cells are also set to the value 0.5. After t_1, agent 1's trust increases and eventually remains at a higher level than agent 2's trust, which can be explained by the fact that agent 1 is better able to detect the NPC since it observes it from the side that has a larger surface area. Another important fact is that agent 2 benefits from agent 1 because he recognizes the same object. This leads

to a higher confidence value, which is why agent 2's confidence level also increases.

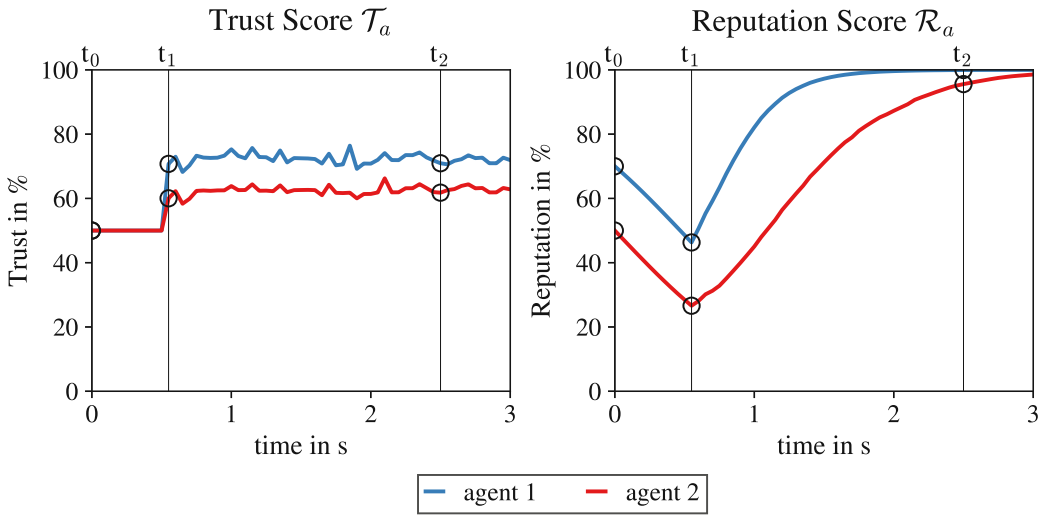

Figure 5.4: Comparison of trust and reputation scores for the two agents 1 and 2.

Looking at the reputation values, we can directly observe the initial reputation of the two agents in the time frame t_0. Since the trust values of both agents are less than 0.55, their reputations decrease in the first period. After time frame t_1, both reputation values slowly increase again and finally reach the highest level.

Finally, the precision and recall values for individual agents are compared to collective perception values in figure 5.5. The results are consistent with the previous observations on trust and reputation. Regarding accuracy, agent 1 has values equal to zero in the first period because it does not recognize any objects. Agent 2 recognizes fairly consistently and is consistent with collective perception since it is the only agent that recognizes in the first period. After time frame t_1, the agent 1 also detects the NPC, resulting in precision greater than zero. Note that precision and recall are a bit noisy, since the recognition algorithm computes a new bounding box for each time frame.
When comparing the recall values, the differences between the agents and the collective perception become clearer: Agent 2 recognizes from the beginning, which is why he has an almost constant recall value in the first period between t_0 and t_1. After t_1, it can be seen that agent 1 achieves a higher recall value than agent 2 because it can recognize the NPC better, which leads to fewer false negative predictions for the grid cells. Agent 2, on the other hand, can only recognize the foremost part of the NPC, which explains the lower hit rate. Also, collective perception has a higher recall than agent 2 in the last period, since it benefits from agent 1's detections. In this case, agent 2 would benefit

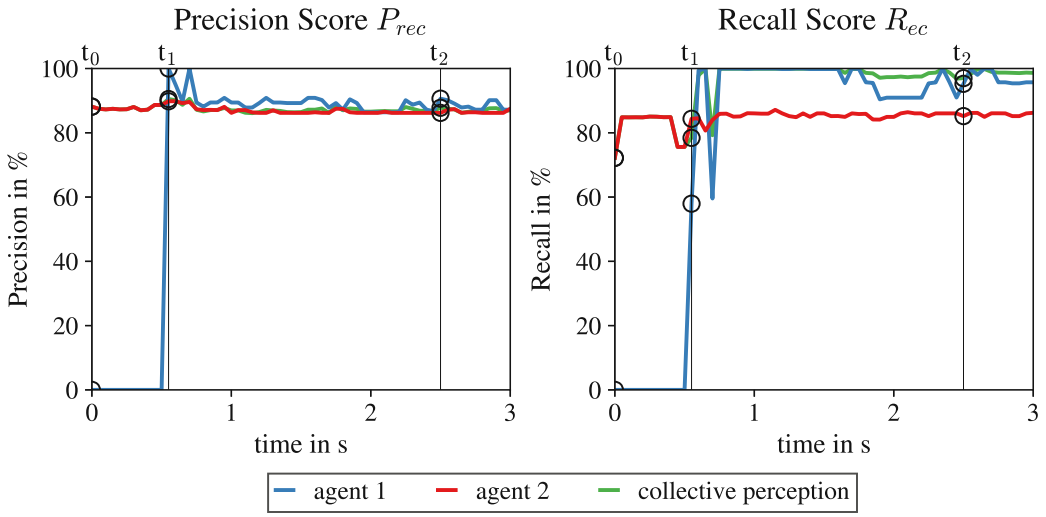

Figure 5.5: Comparison of precision and recall scores of the two agents 1 and 2 as well as collective perception for case 1.

from agent 1 and could use the collective occupancy grid to observe the NPC more closely.

5.2.2 Case 2: No Context Information and High Sun Impact

In the second scenario, weather information is also not considered in the simulation. The measurement confidence $C_{m,a}$ for each agent a remains equal to 1 throughout the simulation. In this case, however, sunlight is assumed to affect the sensors, particularly the RGB camera of agent 2. The sun orientation angles are now chosen to be equal to

$$\alpha_s = 7° \qquad \gamma_s = 87° \tag{5.7}$$

To get an idea of how much the sunlight affects the RGB images, figure 5.6 is shown. It illustrates the images of agent 1 and agent 2 taken in the time frame t_2. It can be seen that agent 2 is very much affected by sunlight, which leads to stray light distortion and blur. It must be mentioned at this point that the lens flare effects were adjusted manually, since CARLA does not change them directly as a function of the sun position relative to the FoVs. For this purpose, the two parameters lens flare intensity and bloom intensity are set to 2.0 and 15.0, respectively.

First, the occupancy probabilities of the agents are compared with collective occupancy probabilities in figure 5.7. It can be seen that agent 1 detects the NPC similar to scenario 1. It is not affected by the sunlight, resulting in high measurement confidence

(a) **(b)**

Figure 5.6: RGB images captured by agent 1 (a) and agent 2 (b) in time frame t_2 for scenario cases 2 and 3. The red boxes mark detections from YOLO v7.

and higher occupancy probabilities. Agent 2, on the other hand, is strongly affected by sunlight because the relative tilt and heading angles are comparatively low. The detection algorithm has difficulty in detection, as can be seen in figure 5.6. It even detects an object on the roof of a building, which can also be seen in the occupancy grids in figure 5.7. In addition, agent 2 does not recognize agent 1, as can be seen in the corresponding figures.
Although agent 2 detects objects poorly, it is certain of them, which is reflected in high detection confidence values and agent occupancy probabilities. The problem of certainty about fictitious objects becomes clearer when comparing the performance values precision and recall.

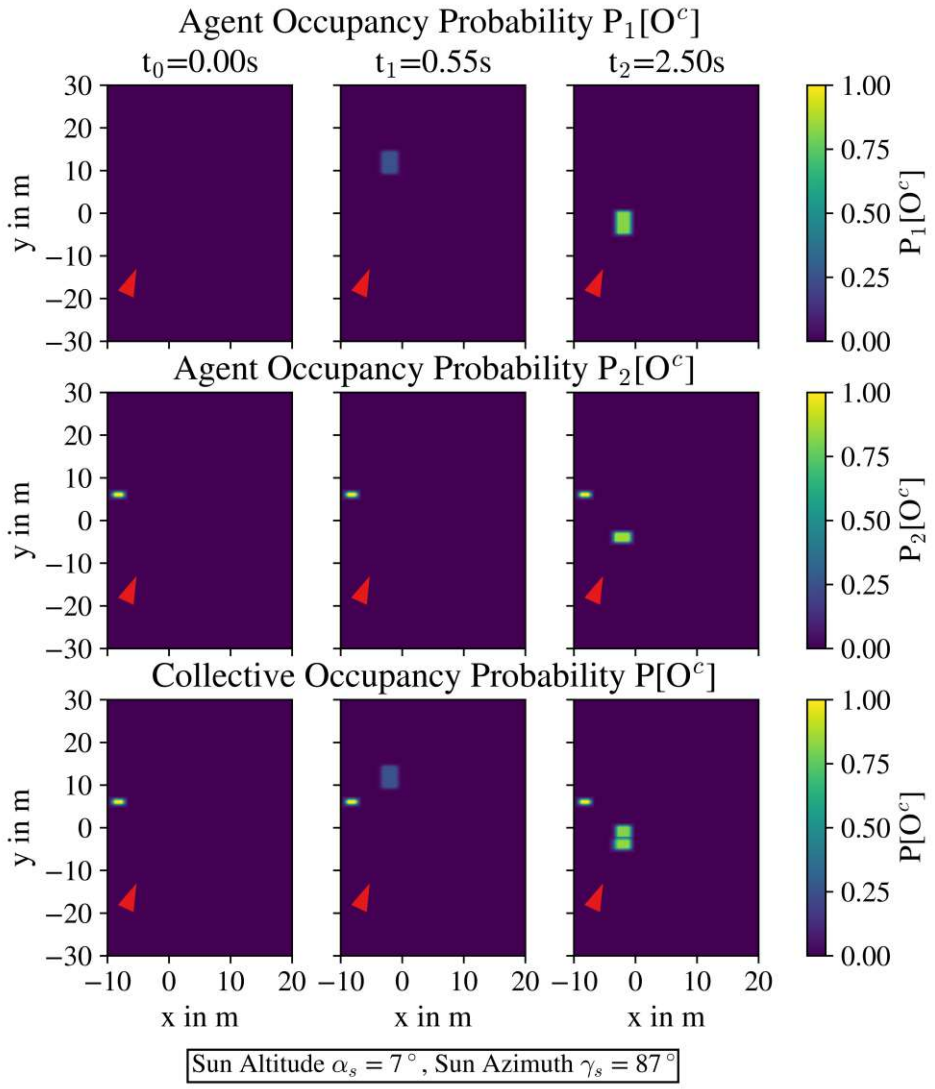

Figure 5.7: Comparison of agent occupancy probabilities and collective occupancy probabilities for all grid cells, the three time frames t_0, t_1, t_2 and scenario case 2. Hereby, the RSU is marked as a red triangle.

Next, the trust and reputation values of the agents are shown in figure 5.8. It can be observed that the trust values of the two agents are very similar throughout the simulation. This phenomenon can be explained by the fact that agent 2 recognizes more objects than agent 1 and is certain of them (agent occupancy probabilities higher than $\kappa = 0.5$ lead to the cell being evaluated as occupied). This would theoretically lead to agent 1 having a lower trust value at the same reputation level. Since the reputation for agent 2 is initially set lower than for agent 1, the effects compensate

each other, resulting in correlated trust values. This effect is particularly evident in this scenario because only two agents rate each other and there is no other agent with whom they share the same opinion than with Agent 1 or Agent 2. In general, however, agent 2's trust scores are relatively high, which is a poor indicator because it makes poor discoveries and should therefore be rated lower than agent 1.
Another point visible in the figure is that the trust and reputation values do not change directly in the time frame t_1. This can be explained by the fact that it is more difficult for the detection algorithm to identify in low light conditions. This leads to a lower detection probability and an agent occupation probability that is eventually lower than κ.

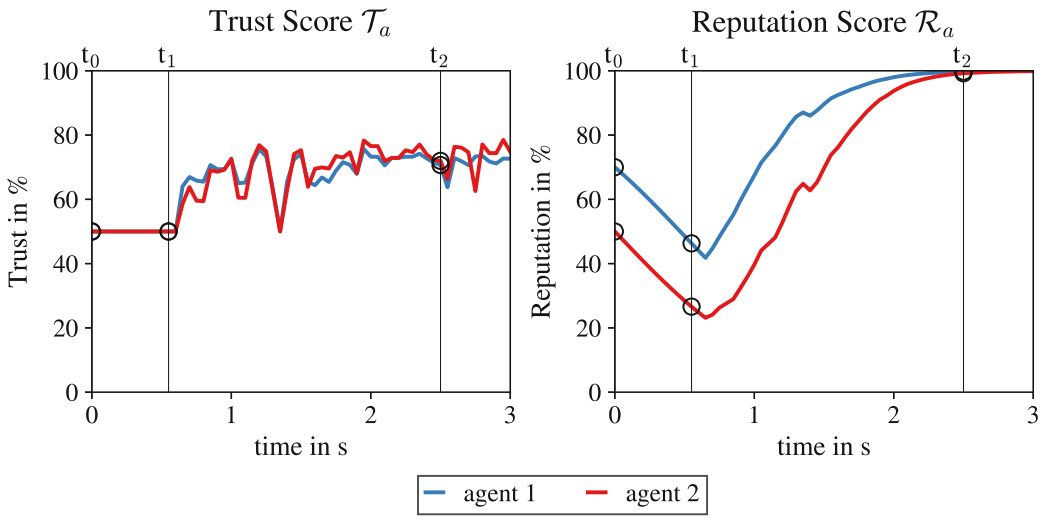

Figure 5.8: Comparison of trust and reputation scores for the two agents 1 and 2.

Finally, in figure 5.9 one can look at the evolution of the precision and recall values of the two agents and collective perception. Looking at the precision values, the same pattern can be observed for agent 1 as in scenario case 1. It is initially zero as the agent detects nothing, and increases rapidly once it begins to detect the NPC. After time frame t_1, the precision of the collective perception and the two agents is similar.
When comparing the recall scores, it becomes clear that the differences in recall are much larger compared to case 1. In particular, agent 2 performs poorly because it recognizes fictitious objects with high probabilities. Although this also leads to a significant deterioration in recall scores in the collective perception, agent 1 can contribute to an improvement in performance due to its high recall scores. These can be explained by the fact that he recognizes the NPC very well, since he observes him from the side and is hardly influenced by the poor lighting conditions.

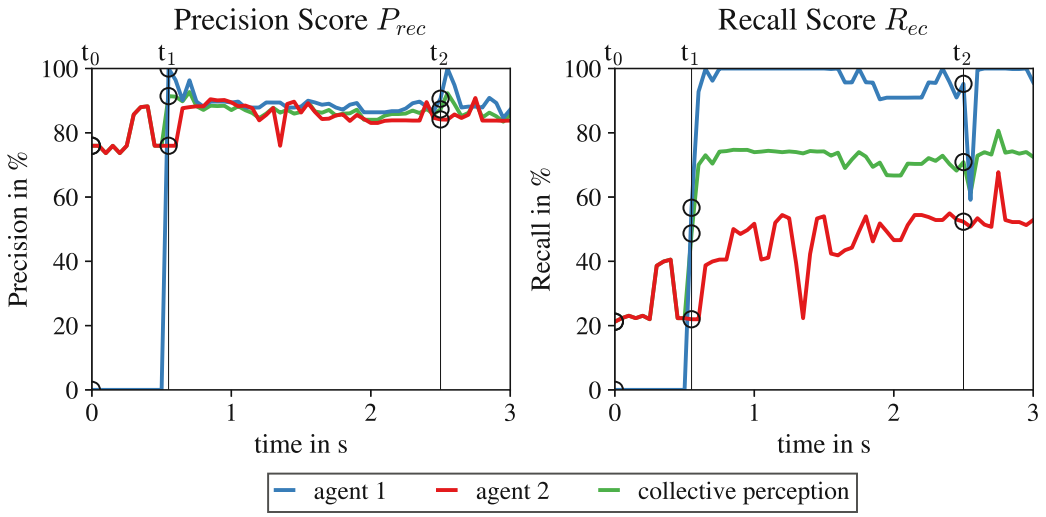

Figure 5.9: Comparison of precision and recall scores of the two agents 1 and 2 as well as collective perception for case 2.

5.2.3 Case 3: Including Context Information and High Sun Impact

In the last scenario, a high sun exposure on agent 2 is assumed again. This time, however, the context information in the form of the solar orientation is taken into account. The measurement confidence values $C_{m,a}$ can therefore now vary between 0 and 1. The angles of the solar orientation are again set to:

$$\alpha_s = 7° \qquad \gamma_s = 87° \tag{5.8}$$

The RGB cameras are affected in the same way as in scenario 2, resulting in blurred images of agent 2, as shown in figure 5.6.
Again, the agent occupancy probabilities are first compared to the collective occupancy probabilities and plotted in figure 5.10. It is directly visible that all agent 2 detections now have lower occupancy probabilities because the measurement confidence values $C_{m,a}$ for agent 2 are set low. This should have the following implications: First, it should lower Agent 2's confidence values, since it now classifies almost all cells as free, while Agent 1 does not. Also, agent 1 has a higher reputation value, which means that its opinion is worth more in the trust evaluation, which just makes agent 2 score lower. Second, it should increase the performance of the collective perception by reducing the number of false positives by lowering the occupancy probabilities of the agents to a level below κ.

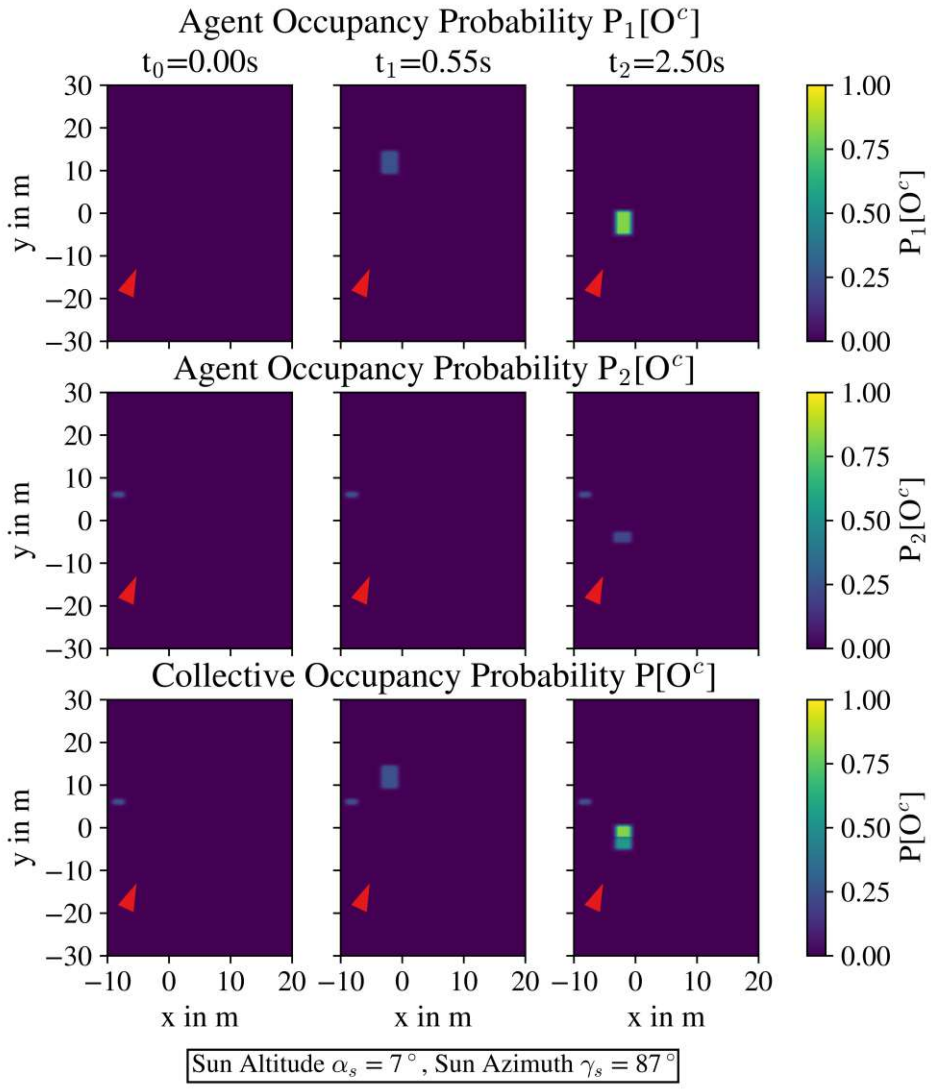

Figure 5.10: Comparison of agent occupancy probabilities and collective occupancy probabilities for all grid cells, the three time frames t_0, t_1, t_2 and scenario case 3. Hereby, the RSU is marked as a red triangle.

Looking at the trust and reputation values of the agents in figure 5.11, the effects described above become clearly visible. The trust values of agent 2 are now lower than those of agent 1 after time frame t_1 due to agent 1's higher reputation and its different observations regarding cell occupancy. Moreover, the reputation of agent 2 now grows much slower compared to case 2, which reflects the lower trust values and is a desired outcome.

Finally, comparing the performance results of the agents with collective perception in

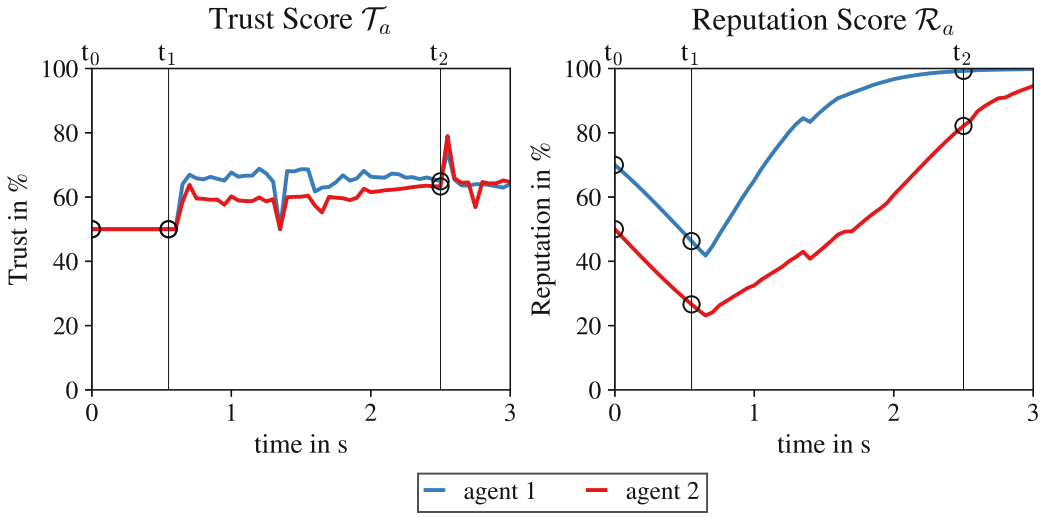

Figure 5.11: Comparison of trust and reputation scores for the two agents 1 and 3.

figure 5.12, one can see similar patterns as in case 2. However, in this case, agent 2's accuracy is close to 100% throughout the simulation, which is due to the lower occupancy probabilities of the agents that do not generate false positives.

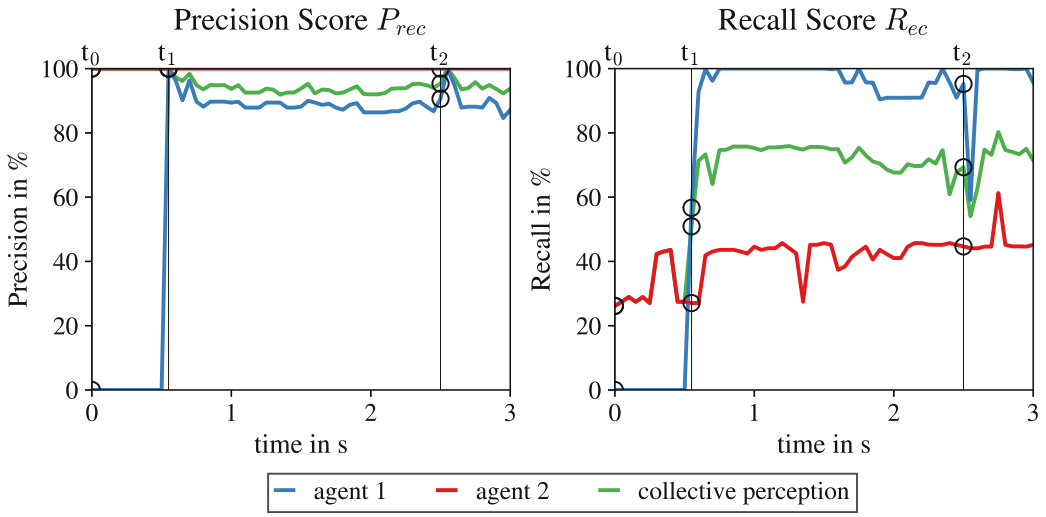

Figure 5.12: Comparison of precision and recall scores of the two agents 1 and 2 as well as collective perception for case 3.

When comparing the recall values, it again becomes apparent that agent 2 performs poorly because it hardly recognizes anything correctly, which leads to many false-

negative cell evaluations. Agent 1, on the other hand, performs much better because it is not affected by sunlight and recognizes the NPC quite well.

5.2.4 Performance Comparison

In this section, the performance of the two agents is compared to the collective perception. The comparison is done by analyzing the F_2-Scores, where a higher F_2-Score means an overall better performance during the simulation. The results are shown in the 5.3 table for each agent, collective perception, and the 3 scenarios. Ideally, the collective perception should perform better than the agents alone for each case, since it benefits from the readings of both agents.

F_2-Score	Agent 1	Agent 2	Collective Perception
Case 1	77.3%	85.3%	93.6%
Case 2	77.4%	48.7%	67.2%
Case 3	77.4%	46.6%	69.0%

Table 5.3: Comparison of F_2-Scores of individual agents, collective perception and scenario cases.

Looking at scenario case 1, it is clear that the collective perception outperforms both agents as it has more information at its disposal and can produce a more accurate occupancy probability map than each of the agents individually. In addition, Agent 2 is overall more performant than agent 1, as it can detect the objects from time frame t_0, while agent 1 can detect the NPC only after t_1.
Comparing the F_2-Scores in the case 2, it can be seen that agent 2 scores significantly worse compared to case 1. This seems to be exactly the influence of the sun on the agents' measurements. Consequently, the collective perception naturally performs worse than in case 1. Agent 1, on the other hand, remains at almost the same performance level, which may be explained by the fact that the sun only affects agent 2.
Finally, comparing the results of case 3, similar patterns can be seen. Agent 2 now performs even worse because it is downgraded by taking the weather context into account. However, this downgrading does not seem to affect the performance of the collective perception in the same way as in case 2. In this case, the collective perception seems to improve by downgrading agent 2. This results in the collective perception in case 3 performing better overall than in case 2.
The results from table 5.3 show that taking the weather context into account is useful and leads to an improvement in collective perception. However, the improvement in this case is quite small, which can be explained simply by the fact that only a small

amount of data is available since the scenario is designed to be as simple as possible. In larger scenarios, this effect should be amplified, and the downgrading of agents by peers becomes easier.

Chapter 6

Conclusions and Future Development

Autonomous driving is on the verge of becoming a reality, but safety concerns need to be addressed. Collective perception technology could be of great benefit to the industry, but its implementation is challenging due to the poor detection performance of some vehicles. Trust management models have been developed to filter out biased detections and improve information quality. However, weather conditions remain a challenge that has not been addressed by existing trust management models.

In this thesis, a grid-based trust management model is proposed to account for the poor detection performance in bad weather. The model considers sun orientation as contextual information when evaluating agents by applying fuzzy logic. Based on this, trust and reputation values are calculated for each agent to evaluate its detection quality. In addition, the model outputs a cell occupancy grid based on the fused information of all agents.

The model was tested and evaluated using the open-source simulator CARLA. The simulation results show that detections of agents strongly affected by bad weather can be filtered by taking contextual information into account. Compared to the case where the weather context is not considered and bad weather conditions are present, collective perception improves its overall performance. This highlights that taking weather into account does indeed have a positive impact on performance and can help to avoid the use of biased detections.

Even though the results already show that the model works, some improvements can still be made, as described below:

- A major improvement for the model would be to expand the rule base and consider

more parameters when calculating measurement confidence. This leads to an even better filtering of biased detections and could improve the rating process of the model. As an example, a rule base could be generated from a trained neural network.

- Another improvement is the extension of the model to three dimensions. Although the proposed model is theoretically already capable of doing this, it was not considered for this work because it would complicate the evaluation.
- Finally, the detection algorithm could be improved to make the generation of three-dimensional bounding boxes more accurate. In this work, the combination of RGB and depth cameras was chosen because it greatly facilitates the calculation of three-dimensional bounding boxes. In a more realistic case, the use of LiDAR point clouds in combination with RGB camera images could be used.

Bibliography

[1] ETSI TR 103 562 (V2.1.1). Tr 103 562: Intelligent transport system (its); vehicular communications. basic set of applications; analysis of the collective perception service (cps); release 2. 2020.

[2] Sarah Ali Siddiqui, Adnan Mahmood, Quan Z. Sheng, Hajime Suzuki, and Wei Ni. A Survey of Trust Management in the Internet of Vehicles. *Electronics*, 10(18):2223, 2021.

[3] Rodrigo Ayala and Tauheed Khan Mohd. Sensors in autonomous vehicles: A survey. *Journal of Autonomous Vehicles and Systems*, 1:1–16, 2021.

[4] Vemema Kangunde, Rodrigo S Jamisola, and Emmanuel K Theophilus. A review on drones controlled in real-time. *International journal of dynamics and control*, 9(4):1832–1846, 2021.

[5] JA Rojas-Quintero and MC Rodríguez-Liñán. A literature review of sensor heads for humanoid robots. *Robotics and Autonomous Systems*, 143:103834, 2021.

[6] Lotfi A Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

[7] David P Watson and David H Scheidt. Autonomous systems. *Johns Hopkins APL technical digest*, 26(4):368–376, 2005.

[8] Michael Fisher, Louise Dennis, and Matt Webster. Verifying autonomous systems. *Communications of the ACM*, 56(9):84–93, 2013.

[9] Chongzhen Zhang, Jianrui Wang, Gary G Yen, Chaoqiang Zhao, Qiyu Sun, Yang Tang, Feng Qian, and Jürgen Kurths. When autonomous systems meet accuracy and transferability through ai: A survey. *Patterns*, 1(4):100050, 2020.

[10] Tao Zhang, Qing Li, Chang-shui Zhang, Hua-wei Liang, Ping Li, Tian-miao Wang, Shuo Li, Yun-long Zhu, and Cheng Wu. Current trends in the development of intelligent unmanned autonomous systems. *Frontiers of information technology & electronic engineering*, 18(1):68–85, 2017.

[11] Imre Rudas and Tamas Haidegger. Verification, trustworthiness and accountability of human-driven autonomous systems. In *2021 IEEE International Conference on Autonomous Systems (ICAS)*, pages 1–1. IEEE, 2021.

[12] Jun Wei Chuah. The internet of things: An overview and new perspectives in systems design. In *2014 International Symposium on Integrated Circuits (ISIC)*, pages 216–219. IEEE, 2014.

[13] Yinong Chen and Hualiang Hu. Internet of intelligent things and robot as a service. *Simulation Modelling Practice and Theory*, 34:159–171, 2013.

[14] Ibrar Yaqoob, Latif U Khan, SM Ahsan Kazmi, Muhammad Imran, Nadra Guizani, and Choong Seon Hong. Autonomous driving cars in smart cities: Recent advances, requirements, and challenges. *IEEE Network*, 34(1):174–181, 2019.

[15] National Highway Traffic Safety Administration. Automated vehicles for safety | nhtsa, 2020. Last accessed 20 December 20222.

[16] Frank van Praat MA MSc RE. So, here's the problem with self-driving cars, 2021. Last accessed 13 March 2023.

[17] Society of Automotive Engineers. J3016: Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *Standard SAE J3016:2021*, 2021.

[18] International Organization for Standardization ISO 4804. 4804: Road vehicles — safety and cybersecurity for automated driving systems — design, verification and validation. 2020.

[19] European Parliament. Self-driving cars in the EU: From science fiction to reality | News | European Parliament, 2019. Last accessed 29 October 2022.

[20] Joven. China's huawei aims to reach driverless car technology in 2025, 2021. Last accessed 29 October 2022.

[21] Dingkang Wang, Connor Watkins, and Huikai Xie. Mems mirrors for lidar: a review. *Micromachines*, 11(5):456, 2020.

[22] Zoleikha Abdollahi Biron, Satadru Dey, and Pierluigi Pisu. Real-time detection and estimation of denial of service attack in connected vehicle systems. *IEEE Transactions on Intelligent Transportation Systems*, 19(12):3893–3902, 2018.

[23] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multimodal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020.

[24] Zurong Qiu, Yaohuan Lu, and Zhen Qiu. Review of ultrasonic ranging methods and their current challenges. *Micromachines*, 13(4):520, 2022.

[25] Jorge Vargas, Suleiman Alsweiss, Onur Toker, Rahul Razdan, and Joshua Santos. An overview of autonomous vehicles sensors and their vulnerability to weather conditions. *Sensors*, 21(16):5397, 2021.

[26] Gary Johnston, Anna Riddell, and Grant Hausler. The international gnss service. In *Springer handbook of global navigation satellite systems*, pages 967–982. Springer, 2017.

[27] Niels Joubert, Tyler GR Reid, and Fergus Noble. Developments in modern gnss and its impact on autonomous vehicle architectures. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 2029–2036. IEEE, 2020.

[28] Kinza Shafique, Bilal A Khawaja, Farah Sabir, Sameer Qazi, and Muhammad Mustaqim. Internet of things (iot) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5g-iot scenarios. *Ieee Access*, 8:23022–23040, 2020.

[29] Shanzhi Chen, Hui Xu, Dake Liu, Bo Hu, and Hucheng Wang. A vision of iot: Applications, challenges, and opportunities with china perspective. *IEEE Internet of Things journal*, 1(4):349–359, 2014.

[30] In Lee and Kyoochun Lee. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business horizons*, 58(4):431–440, 2015.

[31] European Parlament. Directive 2010/40/EU of the European Parliament and of the Council of 7 July 2010 on the framework for the deployment of Intelligent Transport Systems in the field of road transport and for interfaces with other modes of transport Text with EEA relevance, 2010. Last accessed 29 October 2022.

[32] Zhaojun Lu, Gang Qu, and Zhenglin Liu. A survey on recent advances in vehicular network security, trust, and privacy. *IEEE Transactions on Intelligent Transportation Systems*, 20(2):760–776, 2018.

[33] Hanif Ullah, Nithya Gopalakrishnan Nair, Adrian Moore, Chris Nugent, Paul Muschamp, and Maria Cuevas. 5g communication: an overview of vehicle-to-everything, drones, and healthcare use-cases. *IEEE Access*, 7:37251–37268, 2019.

[34] Deva Darshan. Street image, 2017. Last accessed 21 December 2022.

[35] National Highway Traffic Safety Administration. Nhtsa releases 2020 traffic crash data | nhtsa, 2020. Last accessed 29 October 2022.

[36] Florian A. Schiegg, Ignacio Llatser, Daniel Bischoff, and Georg Volk. Collective perception: A safety perspective. *Sensors*, 21(1):159, 2021.

[37] Lili Miao, Shang-Fu Chen, Yu-Ling Hsu, and Kai-Lung Hua. How does c-v2x help autonomous driving to avoid accidents? *Sensors*, 22(2):686, 2022.

[38] Teresa Schmidt, Ralf Philipsen, Philipp Themann, and Martina Ziefle. Public perception of v2x-technology-evaluation of general advantages, disadvantages and reasons for data sharing with connected vehicles. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 1344–1349. IEEE, 2016.

[39] Philipp Themann, Robert Krajewski, and Lutz Eckstein. Discrete dynamic optimization in automated driving systems to improve energy efficiency in cooperative networks. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 370–375. IEEE, 2014.

[40] Sebastian Thormann, Alexander Schirrer, and Stefan Jakubek. Safe and efficient cooperative platooning. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[41] Yifeng Han, Tomoya Kawasaki, and Shinya Hanaoka. The benefits of truck platooning with an increasing market penetration: A case study in japan. *Sustainability*, 14(15):9351, 2022.

[42] Physical Layer PHY Specifications. P802. 11bb™/d4. 1 10 draft standard for information technology—tele-11 communications and information exchange between 12 systems local and metropolitan area networks—13 specific requirements 14. 2022.

[43] TCITS ETSI. Intelligent transport systems (its); communications; architecture; vehicular communication, basic set of applications, part 4: Operational requirements. *Draft ETSI DTS*, 102:637–4, 2010.

[44] Takahito Yoshizawa, Dave Singelée, Jan Tobias Muehlberg, Stéphane Delbruel, Amir Taherkordi, Danny Hughes, and Bart Preneel. A survey of security and privacy issues in v2x communication systems. *ACM Computing Surveys (CSUR)*, 2022.

[45] 3GPP. 3rd generation partnership project; technical specification group services and system aspects; release 14 description; summary of rel-14 work items (release 14). Standard 3GPP TR 21.914, 2018.

[46] Khabaz Sehla, Thi Mai Trang Nguyen, Guy Pujolle, and Pedro Braconnot Velloso. Resource allocation modes in c-v2x: From lte-v2x to 5g-v2x. *IEEE Internet of Things Journal*, 2022.

[47] ETSI EN 302 637-2 (V1.4.1). En 302 637-2: Intelligent transport system (its); vehicular communications; basic set of applications, part 2: Specification of cooperative awareness basic service. 2019.

[48] Christoph Pilz, Andrea Ulbel, and Gerald Steinbauer-Wagner. The components of cooperative perception-a proposal for future works. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 7–14. IEEE, 2021.

[49] Cees G. M. Snoek, Marcel Worring, and Arnold W. M. Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th Annual ACM International Conference on Multimedia - MULTIMEDIA '05*, page 399. ACM Press, 2005.

[50] Eduardo Arnold, Mehrdad Dianati, Robert de Temple, and Saber Fallah. Cooperative perception for 3d object detection in driving scenarios using infrastructure sensors. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[51] Shizhe Zang, Ming Ding, David Smith, Paul Tyler, Thierry Rakotoarivelo, and Mohamed Ali Kaafar. The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car. *IEEE vehicular technology magazine*, 14(2):103–111, 2019.

[52] S Bertoldo, C Lucianaz, and M Allegretti. 77 ghz automotive anti-collision radar used for meteorological purposes. In *2017 IEEE-APS Topical Conference on Antennas and Propagation in Wireless Communications (APWC)*, pages 49–52. IEEE, 2017.

[53] Jacek Wojtanowski, Marek Zygmunt, Mirosława Kaszczuk, Zygmunt Mierczyk, and Michał Muzal. Comparison of 905 nm and 1550 nm semiconductor laser

rangefinders' performance deterioration due to adverse environmental conditions. *Opto-Electronics Review*, 22(3):183–190, 2014.

[54] Mokrane Hadj-Bachir and Philippe de Souza. Lidar sensor simulation in adverse weather condition for driving assistance development. 2019.

[55] Alebel Arage Hassen. *Indicators for the Signal Degradation and Optimization of Automotive Radar Sensors Under Adverse Weather Conditions*. PhD thesis, TU Darmstadt, Germany, 2007.

[56] Jiying Huang, Shenyong Jiang, and Xiaohong Lu. Rain backscattering properties and effects on the radar performance at mm wave band. *International Journal of Infrared and Millimeter Waves*, 22(6):917–922, 2001.

[57] Elliott D Kaplan and Christopher Hegarty. *Understanding GPS/GNSS: Principles and applications*. Artech house, 2017.

[58] PM Kintner, Todd Humphreys, Joanna Hinks, et al. Gnss and ionospheric scintillation. *Inside GNSS*, 4(4):22–30, 2009.

[59] Ismail Gultepe. Fog and boundary layer clouds: fog visibility and forecasting. 2008.

[60] Muhammad Saleem Awan, Erich Leitgeb, M Loeschnig, Farukh Nadeem, and Carlo Capsoni. Spatial and time variability of fog attenuations for optical wireless links in the troposphere. In *2009 IEEE 70th Vehicular Technology Conference Fall*, pages 1–5. IEEE, 2009.

[61] L Hespel, N Riviere, T Huet, B Tanguy, and R Ceolato. Performance evaluation of laser scanners through the atmosphere with adverse condition. In *Electro-Optical Remote Sensing, Photonic Technologies, and Applications V*, volume 8186, pages 64–78. SPIE, 2011.

[62] Clemens Dannheim, Christian Icking, Markus Mäder, and Philip Sallis. Weather detection in vehicles by means of camera and lidar systems. In *2014 Sixth International Conference on Computational Intelligence, Communication Systems and Networks*, pages 186–191. IEEE, 2014.

[63] Alexander Carballo, Jacob Lambert, Abraham Monrroy, David Wong, Patiphon Narksri, Yuki Kitsukawa, Eijiro Takeuchi, Shinpei Kato, and Kazuya Takeda. Libre: The multiple 3d lidar dataset. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1101. IEEE, 2020.

[64] Peter Radecki, Mark Campbell, and Kevin Matzen. All weather perception: Joint data association, tracking, and classification for autonomous ground vehicles. *arXiv preprint arXiv:1605.02196*, 2016.

[65] Yuxiao Zhang, Alexander Carballo, Hanting Yang, and Kazuya Takeda. Autonomous driving in adverse weather conditions: A survey. *arXiv preprint arXiv:2112.08936*, 2021.

[66] Wenbo Sun, Yongxiang Hu, David G. MacDonnell, Carl Weimer, and Rosemary R. Baize. Technique to separate lidar signal and sunlight. *Opt. Express*, 24(12):12949–12954, 2016.

[67] VS Subrahmanian, Judee K Burgoon, and Norah E Dunbar. *Detecting Trust and Deception in Group Interaction*. Springer, 2021.

[68] Wang Yong-hao. A trust management model for internet of vehicles. In *Proceedings of the 2020 4th International Conference on Cryptography, Security and Privacy*, pages 136–140, 2020.

[69] Adnan Mahmood, Wei Emma Zhang, Quan Z Sheng, Sarah Ali Siddiqui, and Abdulwahab Aljubairy. Trust management for software-defined heterogeneous vehicular ad hoc networks. In *Security, Privacy and Trust in the IoT Environment*, pages 203–226. Springer, 2019.

[70] Elvin Eziama, Kemal Tepe, Ali Balador, Kenneth Sorle Nwizege, and Luz MS Jaimes. Malicious node detection in vehicular ad-hoc network using machine learning and deep learning. In *2018 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2018.

[71] Seyed Ahmad Soleymani, Abdul Hanan Abdullah, Wan Haslina Hassan, Mohammad Hossein Anisi, Shidrokh Goudarzi, Mir Ali Rezazadeh Baee, and Satria Mandala. Trust management in vehicular ad hoc network: a systematic review. *EURASIP Journal on Wireless Communications and Networking*, 2015(1):1–22, 2015.

[72] Farhan Ahmad, Fatih Kurugollu, Asma Adnane, Rasheed Hussain, and Fatima Hussain. Marine: Man-in-the-middle attack resistant trust model in connected vehicles. *IEEE Internet of Things Journal*, 7(4):3310–3322, 2020.

[73] Braden Hurl, Robin Cohen, Krzysztof Czarnecki, and Steven Waslander. Trupercept: Trust modelling for autonomous vehicle cooperative perception from synthetic data. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 341–347. IEEE, 2020.

[74] Sergey Chuprov, Ilya Viksnin, Iuliia Kim, Egor Marinenkov, Maria Usova, Eduard Lazarev, Timofey Melnikov, and Danil Zakoldaev. Reputation and trust approach for security and safety assurance in intersection management system. *Energies*, 12(23):4527, 2019.

[75] Jinsong Zhang, Kangfeng Zheng, Dongmei Zhang, and Bo Yan. Aatms: An anti-attack trust management scheme in vanet. *IEEE Access*, 8:21077–21090, 2020.

[76] Weidong Fang, Wuxiong Zhang, Yang Liu, Weiming Yang, and Zhiwei Gao. Btds: Bayesian-based trust decision scheme for intelligent connected vehicles in vanets. *Transactions on Emerging Telecommunications Technologies*, 31(12):e3879, 2020.

[77] Honghao Gao, Can Liu, Yuyu Yin, Yueshen Xu, and Yu Li. A hybrid approach to trust node assessment and management for vanets cooperative data communication: Historical interaction perspective. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[78] Siri Guleng, Celimuge Wu, Xianfu Chen, Xiaoyan Wang, Tsutomu Yoshinaga, and Yusheng Ji. Decentralized trust evaluation in vehicular internet of things. *IEEE Access*, 7:15980–15988, 2019.

[79] Bashar Igried, Ayoub Alsarhan, Igried Al-Khawaldeh, Ahmad AL-Qerem, and Amjad Aldweesh. A novel fuzzy logic-based scheme for malicious node eviction in a vehicular ad hoc network. *Electronics*, 11(17):2741, 2022.

[80] Md Mahmudul Hasan, Mosarrat Jahan, Shaily Kabir, and Christian Wagner. A fuzzy logic-based trust estimation in edge-enabled vehicular ad hoc networks. In *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8. IEEE, 2021.

[81] Susmita Ray. A quick review of machine learning algorithms. In *2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 35–39. IEEE, 2019.

[82] Sohan Gyawali, Yi Qian, and Rose Qingyang Hu. Machine learning and reputation based misbehavior detection in vehicular communication networks. *IEEE Transactions on Vehicular Technology*, 69(8):8871–8885, 2020.

[83] Dempster AP. Upper and lower probabilities induced by a multivalued mapping. *The Annals of Mathematical Statistics*, 38(2):325–339, 1967.

[84] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[85] Mohammed A Abdelmaguid, Hossam S Hassanein, and Mohammad Zulkernine. Samm: Situation awareness with machine learning for misbehavior detection in vanet. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, pages 1–10, 2022.

[86] Fanwei Huang, Qiuping Li, and Junhui Zhao. Trust management model of vanets based on machine learning and active detection technology. In *2022 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, pages 412–416. IEEE, 2022.

[87] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[88] MC Jayaprasanna, VA Soundharya, M Suhana, and S Sujatha. A block chain based management system for detecting counterfeit product in supply chain. In *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pages 253–257. IEEE, 2021.

[89] Marcella Atzori. Blockchain-based architectures for the internet of things: A survey. *Available at SSRN 2846810*, 2017.

[90] Uzair Javaid, Muhammad Naveed Aman, and Biplab Sikdar. Drivman: Driving trust management and data sharing in vanets with blockchain and smart contracts. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–5. IEEE, 2019.

[91] Fatemeh Ghovanlooy Ghajar, Javad Salimi Sratakhti, and Axel Sikora. Sbtms: Scalable blockchain trust management system for vanet. *Applied Sciences*, 11(24):11947, 2021.

[92] Zhaojun Lu, Qian Wang, Gang Qu, and Zhenglin Liu. Bars: A blockchain-based anonymous reputation system for trust management in vanets. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 98–103. IEEE, 2018.

[93] Jorge Godoy, Víctor Jiménez, Antonio Artuñedo, and Jorge Villagra. A grid-based framework for collective perception in autonomous vehicles. *Sensors*, 21(3):744, 2021.

[94] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[95] Google Earth: "New York." 40°42'57"N and 74°00'31"W. June 30, 2022. February 17, 2023. Last accessed 17 February 2023.

[96] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[97] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[98] Hans Hellendoorn and Christoph Thomas. Defuzzification in fuzzy controllers. *Journal of Intelligent & Fuzzy Systems*, 1(2):109–123, 1993.

[99] Franz Dietrich and Christian List. Probabilistic Opinion Pooling. In Alan Hájek and Christopher Hitchcock, editors, *The Oxford Handbook of Probability and Philosophy*, page 0. Oxford University Press.

[100] Joseph Berkson. Application of the logistic function to bio-assay. *Journal of the American statistical association*, 39(227):357–365, 1944.