Perception Enhancement and Improving Driving Context Recognition of an Autonomous Vehicle Using UAVs

Abderraouf Khezaz, Manolo Dulva Hina, Amar Ramdane-Cherif

Journal of
Sensor and Actuator Networks

MDPI

Article

Perception Enhancement and Improving Driving Context Recognition of an Autonomous Vehicle Using UAVs

Abderraouf Khezaz [1,2], Manolo Dulva Hina [1,*] and Amar Ramdane-Cherif [2]

[1] ECE Paris Engineering School, 37 Quai de Grenelle, 75015 Paris, France
[2] LISV Laboratory, Université de Versailles–Paris Saclay, 10-12 Avenue de l'Europe, 78140 Vélizy, France
* Correspondence: manolo-dulva.hina@ece.fr

Abstract: The safety of various road users and vehicle passengers is very important in our increasingly populated roads and highways. To this end, the correct perception of driving conditions is imperative for a driver to react accordingly to a given driving situation. Various sensors are currently being used in recognizing driving context. To further enhance such driving environment perception, this paper proposes the use of UAVs (unmanned aerial vehicles, also known as drones). In this work, drones are equipped with sensors (radar, lidar, camera, etc.), capable of detecting obstacles, accidents, and the like. Due to their small size and capability to move places, drones can be used collect perception data and transmit them to the vehicle using a secure method, such as an RF, VLC, or hybrid communication protocol. These data obtained from different sources are then combined and processed using a knowledge base and some set of logical rules. The knowledge base is represented by ontology; it contains various logical rules related to the weather, the appropriateness of sensors with respect to the weather, and the activation mechanism of UAVs containing these sensors. Logical rules about which communication protocols to use also exist. Finally, various driving context cognition rules are provided. The result is a more reliable environment perception for the vehicle. When necessary, users are provided with driving assistance information, leading to safe driving and fewer road accidents. As a proof of concept, various use cases were tested in a driving simulator in the laboratory. Experimental results show that the system is an effective tool in improving driving context recognition and in preventing road accidents.

Keywords: autonomous vehicle; UAV; drone; ontology; knowledge base; VLC communication; RF communication; safe driving; logical rules

Citation: Khezaz, A.; Hina, M.D.; Ramdane-Cherif, A. Perception Enhancement and Improving Driving Context Recognition of an Autonomous Vehicle Using UAVs. *J. Sens. Actuator Netw.* **2022**, *11*, 56. https://doi.org/10.3390/jsan11040056

Academic Editor: Javier Alonso Ruiz

Received: 6 July 2022
Accepted: 5 September 2022
Published: 20 September 2022

1. Introduction

As with other fields, transportation technology has seen important developments during the last century. With the advent of more powerful computers and more sophisticated electronics, vehicular advancement has reached a point where it is now possible for a vehicle to drive itself alone, without the need for a human driver (or in other cases, with a human driver but with minimal participation). Currently, there are around 1.4 billion cars in the world [1]. For this reason, a driver should be constantly aware of their surroundings, and exercise caution. This case applies to an autonomous vehicle, which is very often evolving in an environment made up of both connected and nonconnected agents. It is then dangerous to rely solely on the communication grid, which might be compromised, hence the need for a dependable perception system.

As shown in Figure 1, an autonomous navigation process may be grouped into four categories [2]: (i) *Perception* is the step where the vehicle uses its sensors to gather information from its surroundings. (ii) *Localization and Mapping* is the step where the obtained data are processed in order to generate precise knowledge about the environment. (iii) *Decision Making* [3] is the step where its intelligence layer generates an optimal action related to the driving situation. (iv) *Action* is the step where the vehicle physically executes the selected

decision. It is interesting to note that perception is the very first node of the process. This is the proof that without a good grasp of its surroundings, it is difficult for a vehicle to make a safe decision. Indeed, by improving the perception process, we can improve the whole process.

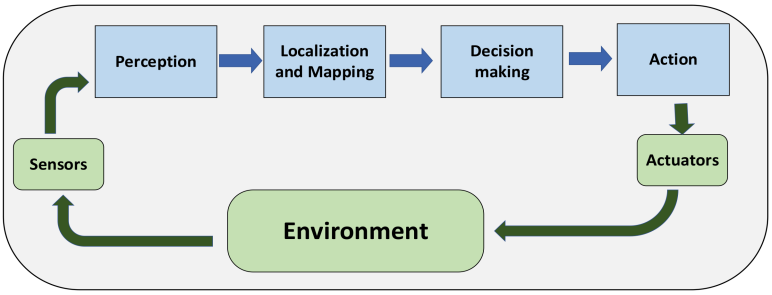

Figure 1. Overview of the autonomous navigation process.

The perception process plays an important role in ensuring the safety of road users. Only when it is provided with enough data can we ensure that an autonomous vehicle can make an optimal choice, providing minimal risk of injuring the driver and passengers, hence the need to make sure that the perception process is reliable.

Figure 2 is an improvement of Figure 1. As shown, a UAV (unmanned aerial vehicle), also known as drone, is used as an extra tool to improve driving environment perception. The UAV sensors provide an external perception, happening in parallel with the local perception of the vehicle. The gathered data are transmitted through a secured communication channel. All the information is then stored in a knowledge base, from where it will be managed and processed.

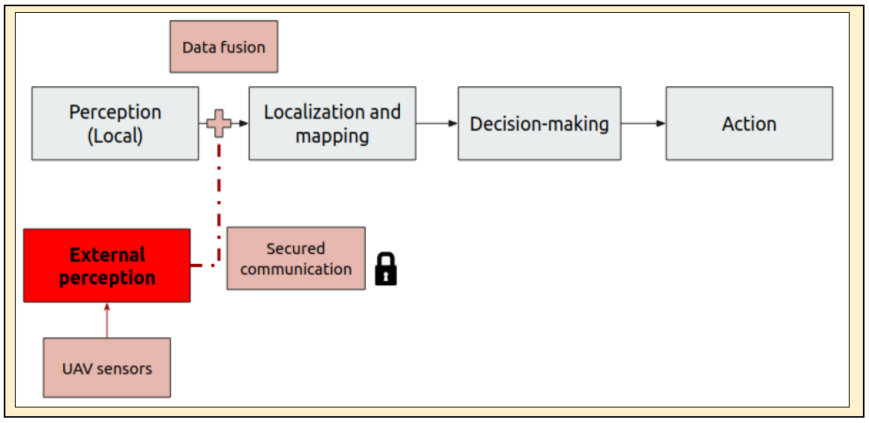

Figure 2. Perception enhancement of the autonomous navigation process.

As shown, the UAV (aka drone) is an extra external sensory mechanism used to gather data and transmit them to the vehicle. The integrity of the exchanged data is guaranteed using a robust communication protocol based on a hybrid VLC/RF protocol. Such data are then stored in the knowledge base, to enhance the perception process in both the local and distant range of a vehicle. The data are processed through a set of logical rules for decision-making purposes. Such a set of rules also includes the management of different entities and communication protocols available. Use cases are presented to validate the above-mentioned concepts.

This work is a continuation of our previous one [4]. Whenever necessary, we will advise the readers to refer to this previous work so as not to repeat discussing here in this paper some materials that were already published.

2. Related Works

Here, we wish to look at the need for and the review of various existing contributions related to perception enhancement. Likewise, we look at how UAVs have been used recently.

Visibility plays a key role in traffic accident prevention [5]. The World Health Organization (WHO) [6] states that "*Seeing and being seen are fundamental prerequisites for the safety of all road users.*" and "*Inadequate visibility is an important factor that influences the risk of a road crash among all types of road users.*" [7]. These criteria are initially aimed at a human driver's visibility but can be easily extended to an autonomous vehicle's ability to perceive its surroundings. The ability to perceive and comprehend the environment is a vital part of an intelligent system, including smart vehicles. In 2018, Van Brummelen et al. [8] made an extensive review of the current state of vehicular perception and defined autonomous vehicle navigation with four main components: Perception, Localization and Mapping, Decision Making, and Vehicle Control.

Perception is using "sensors to continuously scan and monitor the environment, similar to human vision and other senses". To attain this, a good number of relevant sensors may be considered [8–10]:

- *Radar* has been an important sensor in various intelligent transportation applications [11,12]. It is known as a kind of all-weather sensor with high accuracy and long-range sensing capability [13]. It is effective in mid-to-long-range distance measurement with significant accuracy, even in poor weather [14]. It has, however, a small FOV (field of view) with poor performance in near-distance measurement and static object detection. Likewise, its reliability is affected when encountering interference from various sources, including other vehicles.
- *Cameras*, in both single and stereo vision, have lots of potentials. They are the least expensive sensor that can be used [10]. They can be used to (i) monitor the driver's body posture, head position, and eye activity to detect abnormal conditions, signs of fatigue, or the vehicle behaving erratically (driving out of a straight line); and (ii) execute night-vision assistance applications to help drivers see farther down the road and detect objects such as animals, people, or trees in the path that can cause a potential risky situation or accident [15]. They require substantial computational power [16] and may have important costs when used for long-range precise detection [17]. Their performance, however, depends on the weather and brightness [14].
- *LIDAR* (light detection and ranging) continually fires off beams of laser light, and then measures how long it takes for the light to return to the sensor [15]. Its use in various domains has been studied since the 1980s [18], but it is only in the early 2000s that it has found its application in intelligent transportation [19,20]. It is a good tool for 3D mapping and localization with its large FOV [14]. It is dependent on good weather conditions and is less reliable outside its defined range.

Other types of sensors can be found on vehicles, such as ultrasounds. The combination of different types of sensors allows for an efficient perception of the environment. A list of sensors used in traffic control is shown in Table 1.

Considering the current state of vehicular perception, at least five axes of amelioration may be considered [9]:

- Vehicular perception in poor weather and lighting conditions;
- Vehicular perception in complex urban environments;
- Autonomous driving without heavy reliance on a priori perception data;
- Development of safety measures in case of faulty sensors/perception;
- Use of connected vehicle technology to improve accuracy, certainty, and reliability of perception.

As an added tool for perception enhancement, the use of UAVs cannot be underestimated. For example, in 2017, Menouar et al. [21] proposed the use of UAVs as supporting tool for ITS (Intelligent Transportation System). UAVs' abilities to move in a 3D space at high speed, as well as their small size, allow them to transport packages. Drones can be

used to deliver goods, such as in the case of Starship Technologies, which have already made more than 100,000 deliveries as of 2019 [22]. Google and Amazon have already been actively working on UAVs for deliveries, with their respective projects Wing and Prime Air [23,24]. Another UAV use would be for police assistance, patrolling over roads, and looking out for possible offences, such as over speeding or improper lane use. They can register the car's license plate and send information to authorities. However, as pointed out by Sakiyama in her Ph.D. thesis [25], there could be a strong legal and social opposition to this.

Table 1. Sensors used for traffic control (from [15]).

Category	Sensor Type	Application and Use
Intrusive	Pneumatic road tube	Used for keeping track of the number of vehicles, vehicle classification and vehicle count.
	Inductive loop detector (ILD)	Used for detection of vehicle movement, presence, count and occupancy. The signals generated are recorded in a device at the roadside.
	Magnetic sensors	Used for detection of presence of vehicle, identifying stopped and moving vehicles.
	Piezoelectric	Classification of vehicles, counting vehicles, and measuring vehicle weight and speed.
Nonintrusive	Video cameras	Detection of vehicles across several lanes; can classify vehicles by their length and report vehicle presence, flow rate, occupancy, and speed for each class.
	Radar sensors	Vehicular volume and speed measurement, detection of direction of motion of vehicle; used by applications for managing traffic lights.
	Infrared	Application for speed measurement, vehicle length, volume, and lane occupancy.
	Ultrasonic	Tracking the number of vehicles, vehicle presence, and occupancy.
	Acoustic array sensors	Used in the development of applications for measuring vehicle passage, presence, and speed.
	Road surface condition sensors	Used to collect information on weather conditions such as the surface temperature, dew point, water film height, the road conditions, and grip.
	RFID (Radio-frequency identification)	Used to track vehicles mainly for toll management.

The strength of UAVs remains their ability to access spaces that are not easily accessible to vehicles or people. This ease of access can be even more important in the case of being in a dangerous event, such as a leak of dangerous liquid or gas, explosion, or terrorist attacks. Back in 2011, Daniel et al. already proposed the idea of UASs (Unmanned Aerial Systems) for homeland security, deploying a fleet of drones to distribute mobile sensors in incident areas. Given their aerial position, UAVs have a good view of various events happening on a road, in addition to having less impact on traffic. When equipped with a powerful camera, they can easily detect and identify moving cars. For example, in 2017, Xu et al. [26] trained a convolutional neural network (CNN) to locate and follow cars from a height of 100 m to 150 m. Their neural networks showed a car detection success of 98.43%, with robustness on illumination change in moving vehicles. The training time of the algorithm was about 21 h, decent enough for such application; yet the training was only delegated to car detection, without considering other road users (buses, pedestrians, etc.).

So far, UAVs have shown great potential in a vehicular environment. If we are to build a viable model, a communication link between the drones and other actors of the vehicular environment is needed. The integration of UAVs in a V2X protocol has already been investigated by many researchers. In 2018, Hadiwardoyo et al. [27] managed to establish a communication link between a stationary drone and a moving vehicle (see Figure 3). They

experimented on many parameters, such as the height of the drone and the orientation of the transmitting antenna. They managed to reach a transmission ratio of between 60% and 98% over a maximum distance of 2.5 km. They obtained their best results by pointing the antennas vertically and positioning the drone at 100 m, noting that the highest distance allowed by European laws is 120 m.

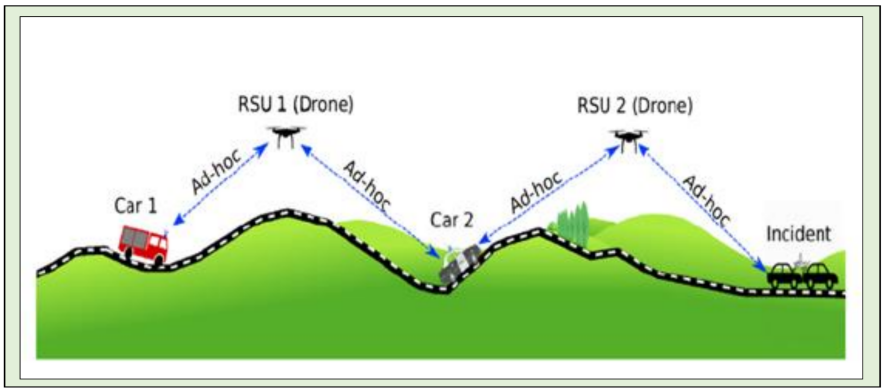

Figure 3. Using UAVs as communication relays [27].

V2X communication is sometimes referred to as VANET (vehicular ad hoc network), the vehicular application of MANET (mobile ad-hoc network). MANETs are described as being "a continuously self-configuring, infrastructure-less network of mobile devices connected wirelessly", meaning that there is no router in the network and the nodes are used for routing instead, despite being in motion and connected to different nodes over time. In the case of VANETs, the nodes of the network are mostly vehicles, but they can also be other infrastructures, such as RSU (roadside unit) or even pedestrians. Studies have already been made focusing on the use of UAVs as the network's nodes. In their review of the applications of UAVs in smart cities, Menour et al. [28] introduced the idea of UAVs being part of a larger network. They proposed a deployment method based on the CDS (connected dominating set) graph theory algorithm, which ensures that all nodes of the same area are always connected by constraining the movements of the UAV (i.e., a UAV that goes too far might break connection with its closest nodes). They could be used as hotspots that can be used for monitoring traffic and for routing information in the network (see Figure 4).

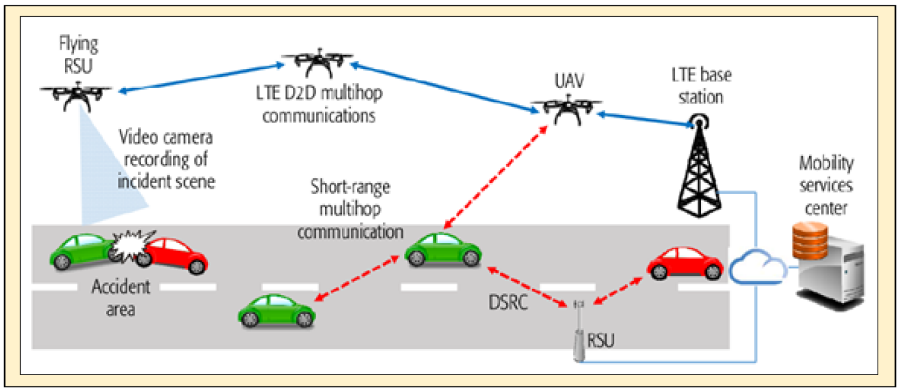

Figure 4. Illustration of UAV/vehicle interactions. Reprinted/adapted with permission from Ref. [28]. 2017, Menouar, H.

Shi et al. [29] made a similar study on the throughput and reached a speed of 1 to 2 MB/s in their simulations with UAVs, comparing regular 802.11 p car-only communications with 2.4 GHz DSRC (dedicated short-range communications) with a swarm of drones. They also noted on the quality of the service, indicating that a higher vehicle density generates a higher delay between messages.

When studying the deployment of a swarm of UAVs, one of its major constraints is its power supply. Currently, most drones have a battery lifetime of 20 to 30 min, and the longer they stay on, the less performant they become. Galkin and DaSilva [30] addressed this issue in their 2018 study and proposed three battery management solutions:

- *UAV swapping*: Having many drones to use and cycling between them. When the active one is low on battery, it returns to the station to recharge and another one takes its place. This solution requires having an important swarm of drones.
- *Battery hot-swapping*: A single drone is used and when its battery is getting low, it returns to the station where an automatic infrastructure quickly changes its battery with a fully charged one. However, this means that the network would be lacking a node for the duration of the battery swap, and this can have some critical repercussions.
- *Wireless power transfer*: In 2018, Ouyang et al. submitted a model that would technically allow to power a UAV with a high-power laser. Unfortunately, this solution has not yet been tested in real conditions, and would require an almost-permanent LOS (line of sight).

The proposed solutions are interesting, but they each have their own limitations. This could be overcome by combining some of them, such as having a dedicated UAV that would serve as an energy refill and make short-time travel to the other UAV that serves as node to recharge it quickly with a laser, given that hot-swapping and a direct physical connections would be critical or difficult to implement. All these considerations highlight the potential of UAVs as dynamic data-gathering tools that can provide useful services to intelligent vehicles. In contrast to the above-mentioned applications, in this paper we will show that UAVs would be useful tools in enhancing driving perception for an autonomous vehicle. To the best of our knowledge, this has not been carried out before, and therefore is the novelty and contribution of this paper.

3. Approach and Methodology

To realize the goal of enhancing perception and cognition of driving context, the following approach and methodology are proposed (see Figure 5).

1. Our ego vehicle is on the road. The vehicle has its own sensors that perceive its environment (e.g., camera, radar, lidar, etc.). The weather, road visibility, and other hazards (e.g., rainy, slippery road, poor visibility, dark road, etc.) evolve with time.
2. Various sensors from the vehicle obtain data from the environment. These data reflect the actual driving situation and different entities present on the road. In Figure 2, this refers to the local perception.
3. The data taken from the environment (both local and external perceptions) are then stored in a knowledge base where they are classified and assigned corresponding properties. In ontology, the individuals of various objects instantiate the generic ontology to form an instantiated ontology.
4. Some missing data are sent to the data-completion process.
5. The data-completion process is undertaken to fill up some missing data. Fuzzy logic rules are also invoked to associate some actual values to fuzzy values (i.e., high, medium, low, etc.).
6. Data fusion is undertaken. The reasoner also cross-references the knowledge base for some objects and how they are involved to some predefined set of rules. This process identifies the logical rules that are applicable to the given situation.
7. The reasoner is the brain of the system's decision-making process. After inferring the situation, the system also identifies the sensors and communication protocols that are appropriate for the current situation. The drones carrying the needed sensors and communication protocols are searched and their activation is requested. They are then deployed as an added layer to enhance environment perception.

A. These collective processes form the simulated environment. This is the visual part that of the system. This is what is visible to the test user during the driving simulation.

B. These collective processes form the decision-making process of the system. As shown, to decide accordingly, the reasoner consults various sets of rules related to the identification of the environment, the selection of correct sensors, the selection of correct entity or UAVs that possess the correct sensors, and the selection of the correct communication protocol.

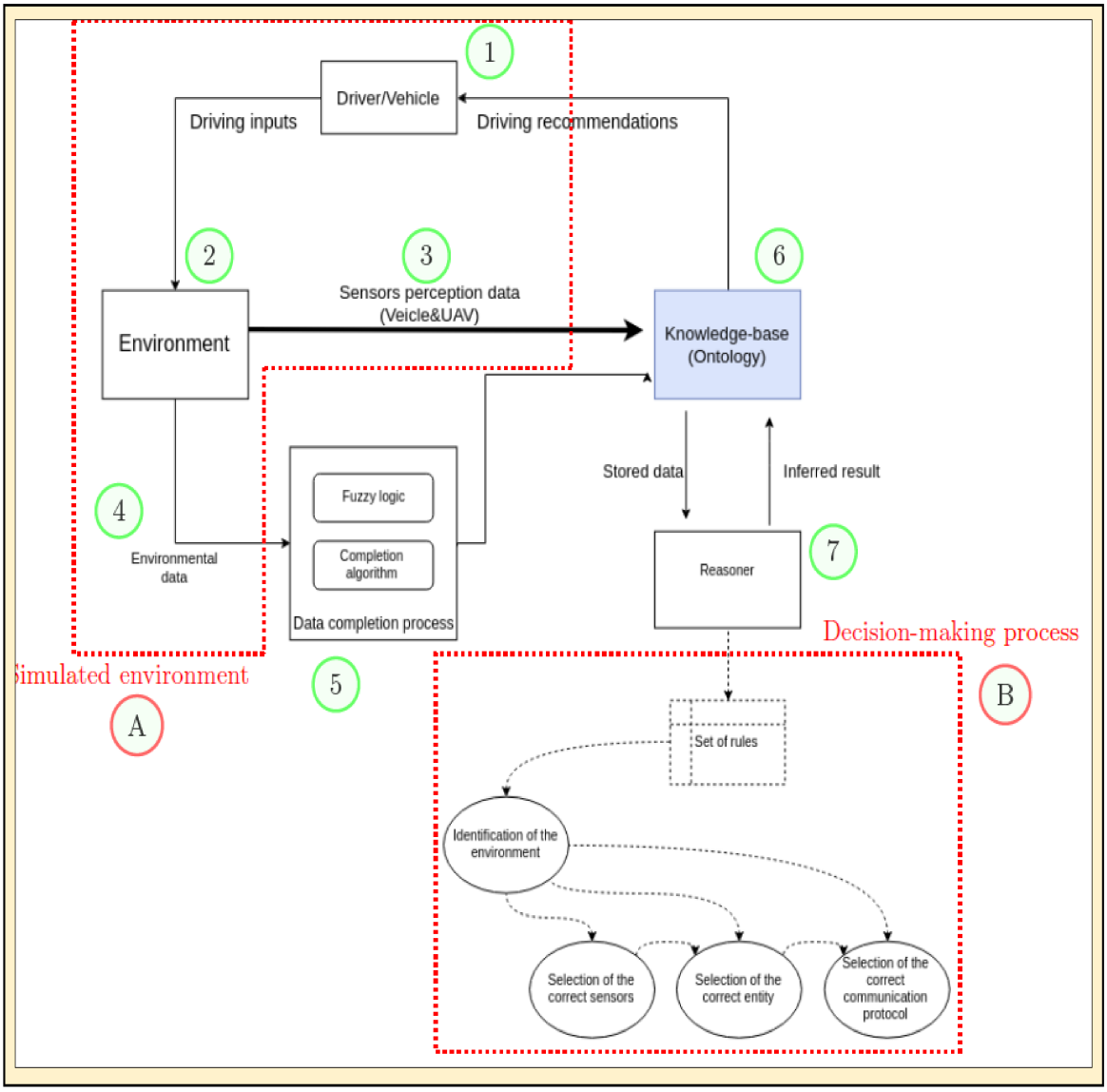

Figure 5. Enumerated approach and methodology used to enhance perception of driving environment.

The process repeats, but this time with UAVs participating to supplement data gathering. In Figure 2, this forms the external perception, which will be fused with local perception to enhance driving context cognition.

Various data of different nature are generated during the driving process; some can be elements related to the ego vehicle, and some are gathered from the vehicle's exteroceptive sensors (e.g., brightness or distance to obstacle). These data may come from the vehicle or from an intelligent agent, such as a drone. These data are stored in an ontology, where they are classified in order to formally represent the environment. This approach offers multiple benefits:

- *Extension of the perception range*: Given their size and positioning, UAVs can cover areas where it would normally be dangerous for the vehicle to venture in.

- *Transmission integrity*: By using two different physical communication protocols for redundancy, the system can make data transmission safe and robust.
- *Proper context identification*: Knowledge bases offer a formal way of describing an environment. When correctly populated, they can be used for an accurate recognition of the driving context.
- *Rigorous management of the enhancement process*: Logical rules make it possible for the reasoner to manage different actors on a scene. When fed with correct rules, the system can quickly determine which sensors and communication protocols should be activated to realize enhanced context perception.

4. Knowledge Representation and Driving Context Detection

In this section, we discuss knowledge representation using ontology [31,32] and a complete ontology, known as a *knowledge base*, and relate them to driving context detection and decision.

A *knowledge base* is an object model way of representing data. As an expert system tool [33], it contains not only all the different actors and entities present in a given situation, but also establishes the abstract concepts, properties, and relationships among stored elements. In addition, a knowledge base is an intelligent layer that can be obtained using inference rules. By setting up an appropriate set of logical instructions, the stored data can be analyzed, processed, compared, and rearranged, producing an output that can be reused. There are different ways to implement a knowledge-based model, such as using logic programming [34], a knowledge graph [35], or an expert system [33]. This work focuses on the use of ontologies.

An ontology serves as a hierarchical data structure containing all the entities of a specific context and the various rules, axioms, and properties governing them. In addition to technical interest, the ontological approach shows some functional features that make it an interesting choice:

- *Scalability*: Once the classes, properties, and rules are defined, the instantiation is managed by a Java API, and it is easy to populate the ontology with new elements.
- *Exportability*: The knowledge base and its actors are independent from the application and can be used for another operation set in a vehicular environment.

The Stanford 101 Guide defines ontology [36] as "*a formal explicit description of concepts in a domain of discourse, properties of each concept describing various features and attributes of the concept, and restrictions on slots*". An ontology basically defines the main actors within a domain of discourses and the different interactions and relationships between them [37]. Ontology [38–40] may be represented by:

- *Classes*: Describe the concepts in the domain, whether they are abstract ideas or concrete entities. Classes can be hierarchized by levels, for example, having a Vehicle as a top class containing Car, Bus, and Bike as subclasses.
- *Properties*: The specific attributes related to a class. They can be intrinsic or extrinsic to an object, representing the interconnections between different concepts.
- *Individuals*: Real instances of classes, representing the elements of the ontology.

An ontology that is complete and filled with a full set of individuals, rules, and properties is referred to as a *knowledge base*. In technical terms, the knowledge base is composed of the Tbox and Abox, which represent the terminological box and assertion box, respectively. The former represents the ontology where the information is stored, and the latter encompasses the rules and properties.

4.1. Data Fusion

Multimodal data fusion is defined by [41] as "the analysis of several data sets such that different data sets can interact and inform each other", meaning that the information from different sources can be compared and cross-referenced to offer a better understanding of the situation where an intelligent agent is evolving. The implementation of this process

requires an architecture capable of efficiently classifying data, and a necessary processing power [42]. In a real driving situation, an important quantity of data needs to be considered, and most of the time, they are of different types. Being able to manage and join seemingly unrelated information could be critical in this context.

As stated earlier, the proposed solution involves using a UAV to gather traffic data and send them back to the vehicle. These data can then be merged with the ones from the vehicle sensors by storing them in an ontology and processing them with logical rules. This concept can also be applied to all other sources of information.

4.2. Reasoning

An ontology allows for the storage and management of different objects from a common context. In addition to being able to represent all the elements of a situation, it is also possible to add a layer of intelligence and reflection through the use of reasoners. A *reasoner* is a tool that can infer logical conclusion from a set of given facts, making the classification of an ontology easier. For example, if we declare V as an instance of a car, and the class Car is a subclass of Vehicle, then the reasoner can infer that V is a vehicle [43].

For a more complex situation, some reasoners can be supplied by logical rules. A rule is a presentation of an Aristotelian syllogism, defined as "a discourse in which certain things having been supposed, something different from the things' supposed results of necessity because these things are so" [44], meaning that if we assume a set of propositions to be true, we can conclude another independent proposition to be also true. Rules are made up of two distinctive parts: an antecedent set of conditions, and a consequent set of actions. They are given in the following form: IF <conditions> THEN <actions>.

There are multiple ways to implement rules into a knowledge base. This work uses the SWRL (Semantic Web Rule Language) approach [45–47]. As stated, it is a language of logic description that enables the combination of different rules to build a more complex axiom. The official documentation gives the following basic example to define the syntax: has Parent(? $\times$ 1, ? $\times$ 2) ^ has Brother(? $\times$ 2, ? $\times$ 3) -> has Uncle(? $\times$ 1, ? $\times$ 3). Here, we are using SWRL notation. The symbol ? means an instance of a class. For example, the axiom Vehicle(?v) means that v is an instance or an individual of the class Vehicle. The has Parent(? $\times$ 1, ? $\times$ 2) is a property related to two objects. It means object $\times$1 has a parent which is object $\times$2. The symbol $\wedge$ means logical AND operator, while -> means a conclusion, a result, or a post-condition. Hence, in the example, by joining the two axioms has Parent and has Brother, it is possible to apply the has Uncle relation to the individuals, hence making the individual $\times$1 the child of $\times$2 and the nephew of $\times$3.

4.3. Fuzzy Logic

Fuzzy logic [48–50] is a method used to handle the concept of partial truth, as opposed to Boolean logic. It is a mathematical means of representing vagueness and imprecise information. A fuzzy logic model is built on the following parts:

- *Meaning representation*: The first step is to translate the given propositions into a set of rigorous and usable constraints. The example given by Zadeh [51] is the following: A fuzzy approach tries to copy a human's ability to infer based on previous experiences and social norms; thus, when we state that "usually Swedes are blond", the term "usually" may be given an accuracy value of between 50% and 80%.
- *Knowledge-based system*: By definition, a knowledge base is a way to store complex structured and unstructured information, making it an "advanced" database. The storing function is then completed by an inference engine, a tool that allows the logical processing and reasoning on the stored elements. This process is enabled using an IF-THEN type of rules, which can particularly be useful when dealing with fuzzy logic.
- *Defuzzification*: The last part of the process is the inverse transformation that is opposite to the first one. It associates the results of the previous rules to a crisp logic equivalent

value. The receiving algorithm requires a real value in order to function and relies on the defuzzification output in order to do so.

By its nature, fuzzy logic allows for better management of uncertainty in engineering and has found a fundamental role in many AI fields due to its potential in dealing with a lack of data.

Fuzzy logic relies on the notion of the *Membership* function: an uncertain statement is represented by a mathematical function covering a value ranging from 0 to 1. When we refer to a sensor as functioning "*Perfectly*" or "*Completely not functioning*", those are the extreme possible value. However, taking uncertainty into consideration means that the meanings of "*Good*" and "*Poor*" should be considered and represented. Membership functions allow for the representation of how reliable an information is: when the *Y*-axis value of a fuzzy logic function is at 1, that is the equivalent of the literal expression "*Always*" or "*Definitely*", while when it is 0, that would mean "*Never*" or "*Definitely Not*". This is a proven way of representing just how certain information can be.

We apply this process to the management of sensor's reliability. Indeed, for many reasons, a sensor's precision can sometimes drop because of internal or environmental issues. A fuzzy logic approach would allow us to combine inputs from multiple potential faulty sources and try to infer a correct state of the environment where the vehicle is evolving. As stated, the most common sensors in autonomous vehicles are (i) monoscopic cameras, (ii) stereoscopic cameras, (iii) infrared cameras, (iv) RADAR, (v) sonar, and (vi) LIDAR. They need to reach a *threshold level of precision*. In addition to internal issues, the other source of uncertainty of sensors comes from brightness and weather states [8]. When considering the sensor's data, it is important to consider a margin of uncertainty. In order to have a better grasp of the environment, various variables are considered and treated, producing a *Perception score*. This parameter will classify the environment in which the vehicle is evolving, which will influence the need for sensor activation.

The membership functions of fuzzy logic allow for the transposition of fuzzy expressions into numerical and exploitable values. Once quantified, the inference and defuzzification processes can happen. In the context of this work, we are interested in the evaluation of the capability to perceive the environment, meaning that Perception will be quantified into the following values: *Poor*, *Average* and *Good*. Logical rules will combine values of different variables to infer a general perception value. The rules are in the form of logical IF-THEN structure, making it easy to combine different variables (see Figure 6). In the first case, the brightness is poor, but the weather is great, making the general perception *Average*. In the second case, both brightness and weather are good, making the perception assessment *Good*. This approach can also be applied to sensors. Examples are shown in Figure 7.

1. IF (brightness IS dark) AND (weather IS good) THEN (perception IS average)

2. IF (brightness IS good) AND (weather IS good) THEN (perception IS high)

Figure 6. Defuzzification rules.

1. IF (cameraMono IS poor) THEN (perception IS poor)
2. IF (cameraMono IS good) AND (NOT(brightness IS good)) THEN (perception IS average)
3. IF (cameraMono IS good) AND (brightness IS good) AND (Weather IS good) THEN (perception IS good)

Figure 7. Defuzzification rules for sensors.

The first rule is a case where the camera sensor is declared poor, meaning it is unreliable due to some malfunction or environmental issues. This situation would automatically impair the perception score. In the second example, the sensor works well but the environment is hindering, making the perception average. In the third example, the sensor functions properly and the conditions are good, making the overall perception good.

The perception scores from different functions are averaged to a single value which will be the general output of the system. If the majority of the functions have a "*Good*" perception value, it would be the returned value to the Perception class.

The *weather* is a parameter that has a value that ranges from "*Sunny*" to "*Rainy*" with numerical associations in the range of [0:10]. It has a fuzziness equivalent of "*Good*" or "*Poor*", depending on what is considered to be "good weather" or "bad weather". The representation is shown in Figure 8. The values were chosen in an arbitrary way of what a "good" or a "poor" value means. The chart has two different functions: one for "*poor*" and one for "*good*". As stated, when the weather value is close to 0, the environment is sunnier, which is universally considered to be the best weather, as well as the optimal situation for visual perception. The "*Good*" function is at the max value for 0, which is a perfect sunny weather, and starts to slowly decrease proportionally to the weather, reaching a null value for bad weather. On the other hand, the "*Poor*" function only starts to rise at the value of 6, which would consist of a cloudy or light rain situation (0 equals sunny, while 10 equals rainy). At the max value, the membership function is at 1, meaning that a rainy situation is indeed considered bad weather.

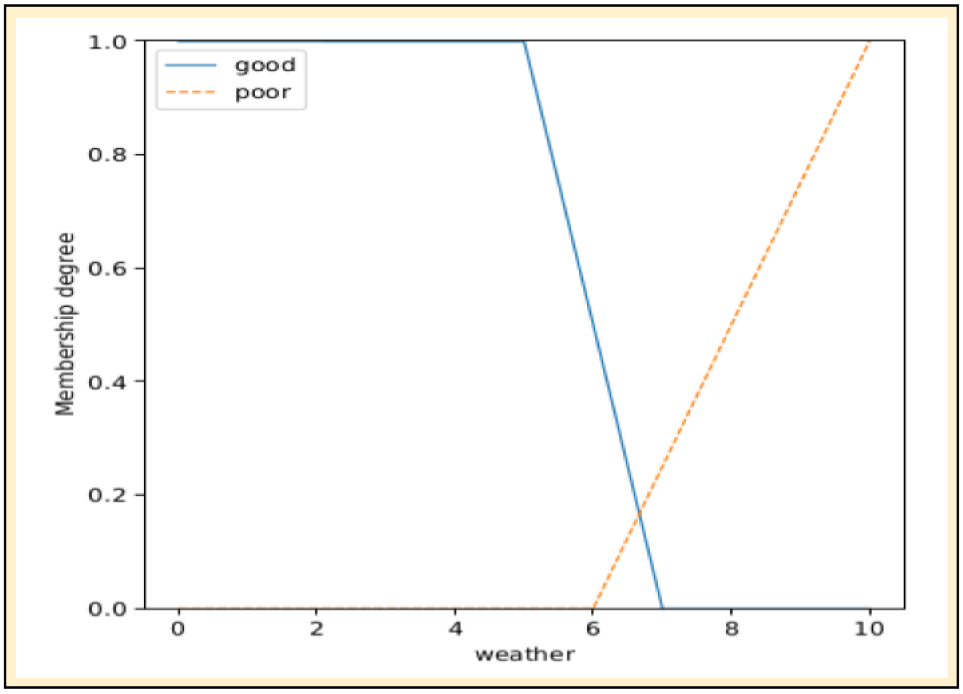

Figure 8. The fuzzy representation of weather.

Brightness is a value that ranges from "*Dark*" to "*Overbrightness*" with numerical associations in the range of [0:10] (see the fuzzy representation of brightness in Figure 9). It has a fuzziness equivalent of ["*Dark*", "*Good*", "*Bright*"], with an optimal brightness achieved at 5. The lowest values correspond to Darkness and the highest to Overbrightness, both having a negative effect on perception. The membership values for Brightness are split between three functions: "Dark", "Good", and "Bright". The lower the brightness, the higher the "Dark" function membership value is, while too bright an environment is considered as too "Bright", meaning overbright.

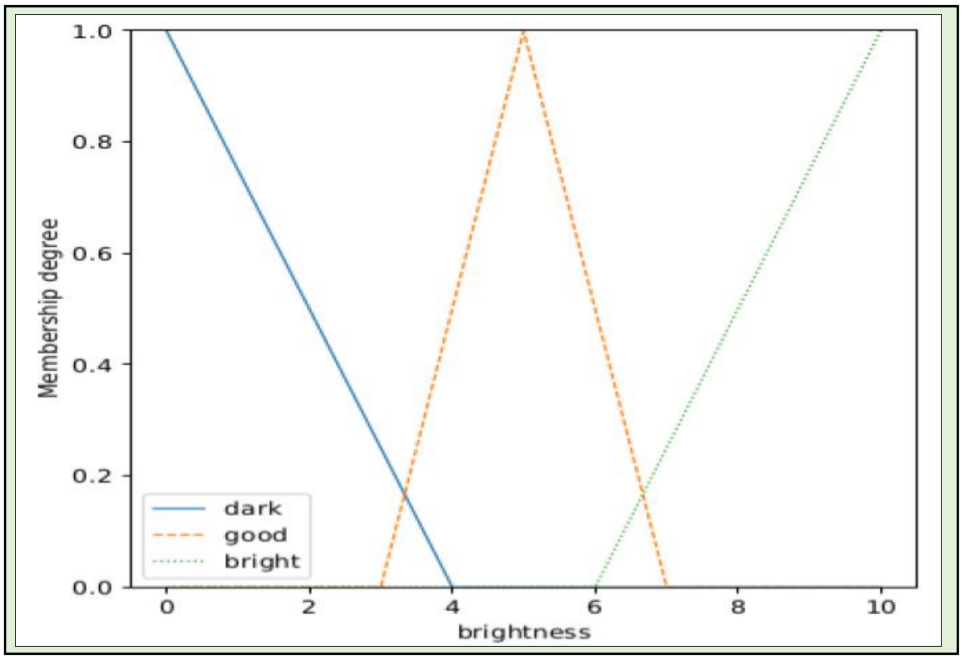

Figure 9. The fuzzy representation of brightness.

Sensors also have their own membership functions, but they all are subdivided into "Good" and "Poor" reliability. This is due to weariness, environment, quality of the sensor, or undervoltage. For example, a stereoscopic camera, which is more expensive and of higher quality, will be required to have a stricter definition of "good functioning" than a sonar, a cheaper and more common component. The details of each membership functions are shown in Figure 10. The *X*-axis defines the quality of the sensor, a variable depending on elements such as the sensor current state or maintenance routine. The higher this value is, the more confident we can be on the reliability of the gathered data. This is represented by a higher membership value. The difference in the functions is related to the type of sensor. Some sensors can generate more or better-quality data, hence the need to be stricter in the way they are trusted.

4.4. Knowledge Base Implementation

All the data stated earlier are treated in order to generate an output called *PerceptionAccuracy*, whose value varies from [0:3], with a higher score representing a better Perception. This is later logged in the knowledge base. In the knowledge base, the Perception score is classified as Poor, Average or Good. Its value has an impact on how trusted the environmental detection is.

As stated, an ontology serves as the storage of different elements, making up a specific context as well as their properties and interlinked relationships. An important number of elements classified will naturally result in a more precise ontology. The data set can be supplemented through completion algorithms (not discussed in this paper; see [52] for more details), or fuzzy logic methods can be applied in order to improve the certainty of the perception. This approach is applied to the environment detection.

4.4.1. Sensors

The Sensors class in the ontology includes all the sensors that can be used in a road environment. They are subdivided in different subclasses, as shown in Figure 11:

- *Active Sensors*: Sensors that rely on their own source of emission in order to gather data. Lidars use a laser ray in order to map their surroundings. Radars generate a high-frequency electromagnetic impulse and use the Doppler effect [53] in order

to calculate the distance. Ultrasound sensors rely on the same principle, but with ultrasonic waves.

- *Passive Sensors*: Sensors that gather data without the need for generating any form of emission. They mostly refer to cameras. There are three different types of cameras: (i) monoscopic cameras, the classical and most common ones, used for classification and image processing; (ii) stereoscopic cameras, which rely on the same principle except with more cameras, allowing for depth perception; and (iii) thermal cameras, cameras using infrared radiation, which allow for their use in bad weather and poorly illuminated environments but make the detection and classification of elements harder.
- *Environmental Sensors*: Sensors used for the detection of environmental variables, including rain, fog, and brightness.

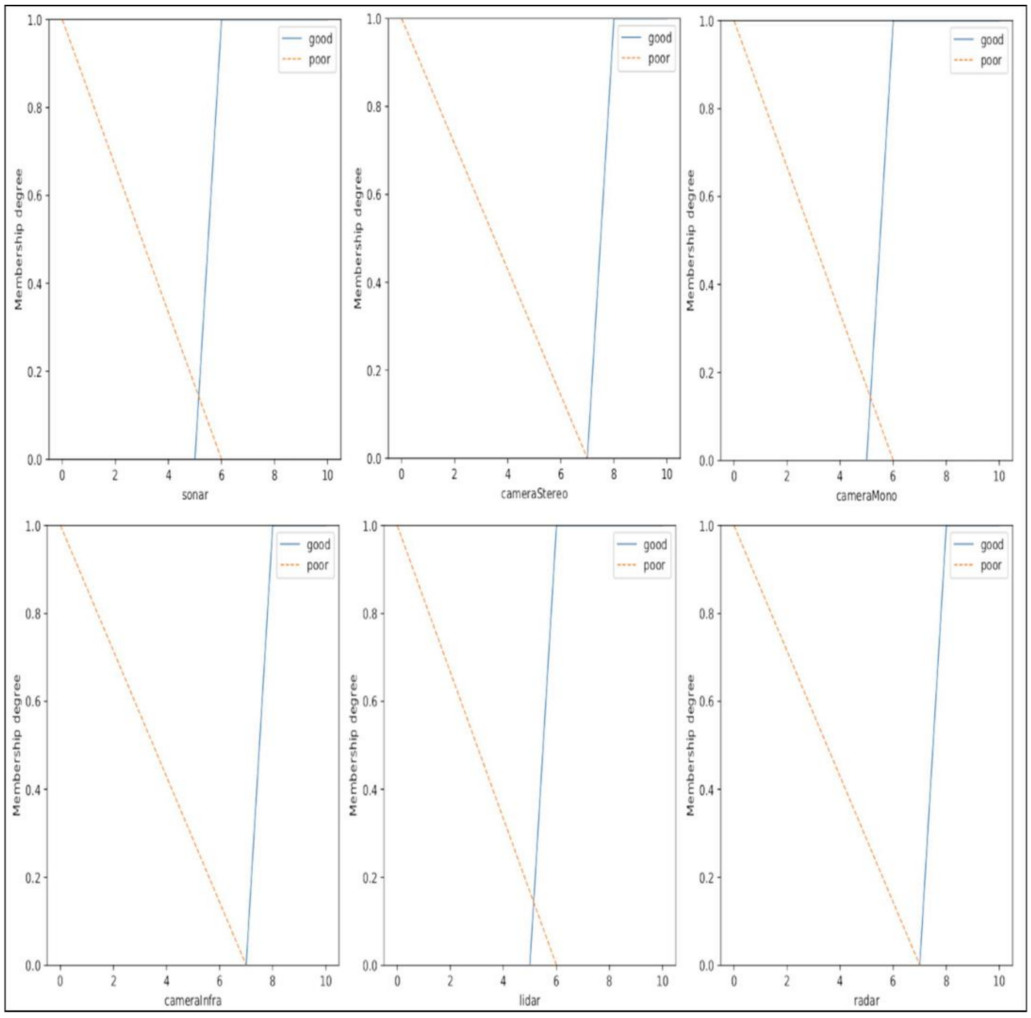

Figure 10. The fuzzy representation of sensors.

More details about the sensors can be found in the representation in Figure 12, especially concerning their different properties:

- *isWeakToRain* property means that the sensor performs poorly in the case of rain;
- *isWeakToFog* property means that the sensor performs poorly in the case of fog;
- *isWeakToBrightness* property means that the sensor performs poorly in the case of overbrightness;
- *isWeakToDark* property means that the sensor performs poorly in the case of lack of illumination;
- *isUnreliable* property means that the sensor performs poorly due to some reasons;
- *hasMinRange* property represents the minimal functioning range of a sensor;

- *hasMaxRange* property represents the maximal functioning range of a sensor;
- *isActiveSensor* property defines if a sensor should be activated or not depending on the environmental situation.

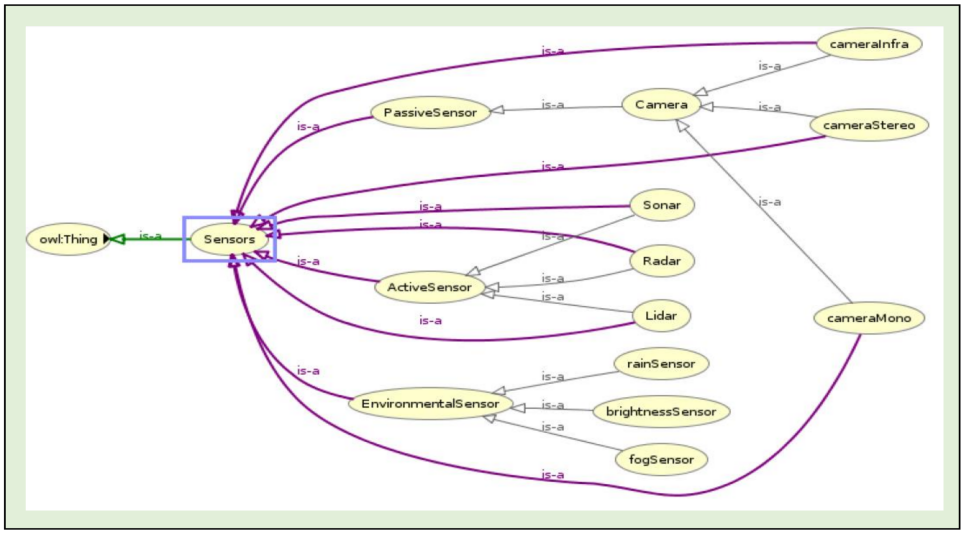

Figure 11. Relationships and properties of the Sensors class in the ontology. Following the notion of object-oriented programming, the arrow here denotes inheritance.

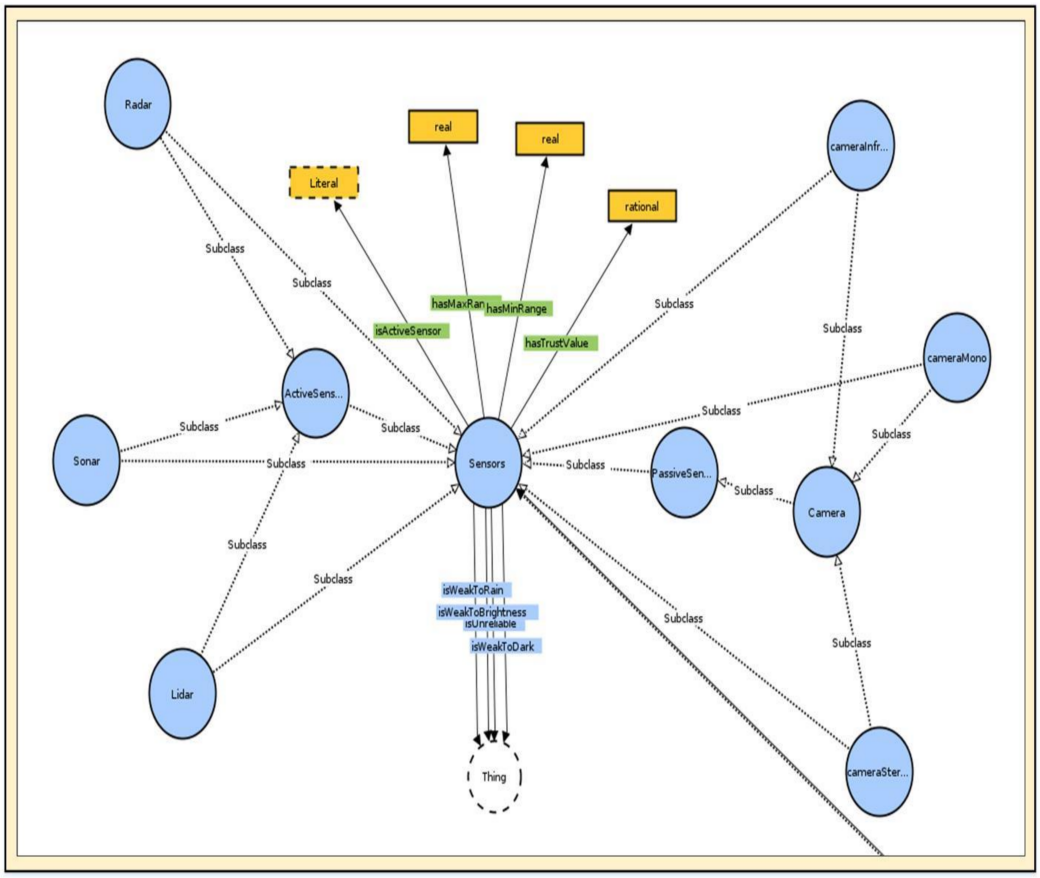

Figure 12. Relationships and properties of the Sensors class.

4.4.2. Vehicle

The Vehicle class includes all the different vehicles that can be encountered in a road environment, including cars, trucks, and bikes, as well as UAVs. Here are the details of some properties of the class:

- *hasDriver* is a functional property between a vehicle and a driver. It allows for the identification of the ego vehicle, which will contain the main driver individual.
- *hasSensor* is a property that associates a vehicle with all the sensors embedded on it.
- *hasDistanceFromVehicle* is a property that allows for the fuzzy classification of the distance between a vehicle and a physical obstacle on the road. It can have a value of "*Far*", "*Medium*", or "*Close*". It can be associated with the Boolean data property that has obstructed view.
- *isCloseToUAV* applies when the ego vehicle is within the communication reach of a UAV. If the conditions are met, it can be associated with the Boolean data property that is active UAV for the data-gathering request.

4.4.3. Communication Protocol

The communication protocol involved in this paper is the one involved in the data transmission between a UAV and a vehicle.

As stated, this paper is a continuation of a previous one [4]. To avoid repeating materials on communication protocols that were already published, readers are advised to refer to it whenever they feel that some discussions in this manuscript are insufficient.

The *CommunicationProtocol* class in the ontology includes three possible disjoint communication protocol classes: RF, VLC, and hybrid. The only notable property of this class is *isActiveCommunicationProtocol*, which is a Boolean value, indicating which communication protocol should be used depending on the situation. Protocols function differently depending on the environment, and the hybrid approach, when available, allows for a reinforced data transmission.

Whenever a UAV is equipped with both RF and VLC technologies (making it possible to implement hybrid protocol), and on the condition that the environment has no particular hazard, then the system may opt for the activation of the hybrid protocol. This reinforces communication, thus allowing safer data communication. In a hybrid setting and to ensure that no information is lost during data transmission between two agents, the "*heavy*" data is sent through VLC, due to its high-speed and reliability; and the *hash of the data* (much smaller data, used to verify the integrity of the transmitted information) are sent through an RF channel.

Some events happening on a road may negatively impact RF communications, such as a fire hazard. In such a case, RF is not the appropriate communication protocol. In a foggy weather, VLC (visual light communication) is not the appropriate communication protocol. Indeed, the choice of the suitable communication protocol is a function of the weather and fire/heat hazards.

4.4.4. Logical Rules

Once the ontology is populated with individuals, the elements in the knowledge base are processed using a reasoner and a set of rules. Rules follow the logical IF-THEN pattern. Reasoners work in a concurrent way, meaning that the rules are executed simultaneously and not sequentially. However, for readability and clarification purposes in this manuscript, they have been regrouped in categories based on their common goal.

4.4.5. Environmental Detection

Identifying the correct environment is a critical part of the process. After the data reinforcement process, the object fed to the knowledge base can be used in order to correctly infer the environment. This is carried out thanks to logical rules.

- The environmental sensors gather the value of an environment variable and compare it to a threshold IF (envSensor exists for environment X) AND (envSensor returns envValue) AND (envValue bigger than threshold value) THEN (environment is of type X).
- In the case that different contexts can be linked to a single environment (for example, a rainy or foggy context both correspond to a bad weather situation), then this can also

be simplified by using logical rules IF (environment is Rain) THEN (environment is BadWeather); IF (environment is Fog) THEN (environment is BadWeather).
- Some other situations can be covered, for example, limited visibility IF (object on the road) AND (vehicle on the road) AND (object is very close to the vehicle) THEN (the view is obstructed) or a fire hazard event IF (vehicle on the road) AND (building nearby) AND (building is on fire) THEN (there is a hazard of type FireHazard) AND (environment is of UnusualEnvironment type).

4.4.6. Entity, Sensors, and Communication Protocols Management

The proposed model is aware of what entity is in the vicinity of the vehicle, and given the context of the environment, can also infer what sensors and communication protocols are appropriate for the specified context. All these can be managed using the logical rules.

- When a UAV is identified, there is a need to check if it contains the correct sensors: IF (there is a vehicle) AND (there is a UAV) AND (there is an environment of type X) AND (there are sensors appropriate for that specific environment) AND (the UAV is carrying the sensors) AND (the UAV is close enough to the vehicle) THEN (the UAV is activated). Activating a UAV in the knowledge base means requesting perception data from it.
- Once activated, there is also a need to activate the needed sensors: IF (there is a UAV) AND (there are sensors working on that specific environment) AND (the UAV is carrying the sensors) AND (the UAV is active) THEN (activate the sensors on the UAV).
- The same methodology is applied to the communication protocol: IF (there is an environment of type X) AND (there are communication protocols appropriate for environment X) AND (there is a UAV) AND (the UAV can communicate using the defined protocols) THEN (activate the communication protocols).

As stated, due to the nature of a reasoning engine, these different tasks are actually implemented in parallel. However, for readability purposes, the rules are regrouped in a modular way in this paper.

4.5. Decision-Making Process

The previous sections introduced the different elements of the model as well as provided an introduction to the logical rules used for this work. This section will dive deeper into the SWRL rules covering different situations.

4.5.1. Detection of Environment with Respect to Weather and Other Hindrances

In order to deal with the varying status of weather, different rules exist in order to identify the situation that the vehicle is currently evolving in. Due to the importance of a correct environmental hazard detection, there is an additional *PerceptionAccuracy* variable that decides if the data obtained from the environment can be trusted based on the general state of perception when it is considered good enough.

The values from the environmental sensors are gathered and processed using the rules shown in Figure 13. Using the environmental sensors, if the detected value is above a certain threshold, the weather status is changed to the detected one (Fog, Rain, or Snow).

The set of rules in Figure 14 rely on the Weather variable to infer the Environment status. The default "Sun" value results in a *Normal* environment, while the other status will lead to a *BadWeather* environment.

A similar approach is taken for the identification of brightness level of an environment. The associated rule found in Figure 15 allows for the detection of a dark environment, which can influence the driver's visibility while driving on the road.

```
fogSensor(?fogS) ^ hasFogValue(?fogS, ?fogV) ^ PerceptionAccuracy(?P)^
Good(?P)^ swrlb:greaterThan(?fogV, 50) ^ Weather(?W) -> Fog(?W)

rainSensor(?rainS) ^ hasRainValue(?rainS, ?rainV) ^ PerceptionAccuracy(?P)^
Good(?P)^ swrlb:greaterThan(?rainV, 50) ^ Weather(?W) -> Rain(?W)

snowSensor(?snowS) ^ hasSnowValue(?snowS, ?snowV) ^ PerceptionAccura-
cy(?P)^ Good(?P)^ swrlb:greaterThan(?snowV, 50) ^ Weather(?W) -> Snow(?W)
```

Figure 13. Detection of weather using sensors.

```
Sun(?S) ^ Environment(?Env) -> NormalEnvironment(?Env)
Fog(?F) ^ Environment(?Env) -> BadWeather(?Env)
Snow(?S) ^ Environment(?Env) -> BadWeather(?Env)
Rain(?R) ^ Environment(?Env) -> BadWeather(?Env)
```

Figure 14. Classification of the environment based on the weather.

```
brightnessSensor(?brightS) ^ hasBrightnessValue(?brightS, ?brightV)
^ swrlb:lessThan(?brightV, 50) ^ Environment(?Env) -> Dark(?Env)
```

Figure 15. Detection of brightness using sensors.

4.5.2. Entity and Sensor Activation

Once an environment has been correctly assessed, the model needs to ask for perception enhancement. Figures 16–18 illustrate a simple activation scenario: suppose that a vehicle is in a situation wherein the view is obstructed. Assume further that there is a UAV stationed nearby with adequate sensor (in this case, a simple camera). If the vehicle is close to the UAV (the isCloseToUAV condition is marked as True), then the system asks for the activation of the UAV and its sensors. This means the start of data gathering and the transmission of such data by the drone to the vehicle.

```
hasSensor(?U, ?cm) ^ UAV(?U) ^ cameraStereo(?cm) ^ Vehicle(?V) ^
Object(?O) ^ hasObstructedView(?V, ?O) ^ NormalEnvironment(?Env)
^isCloseToUAV(?V,?U) -> isActiveUAV(?U, true) ^ isActiveSensor(?cm, true)

Vehicle(?V) ^ hasDistanceFromVehicle(?O, NearDistance) ^
Environment(?Env) ^ Object(?O) -> hasObstructedView(?V, ?O)
```

Figure 16. Obstructed view.

```
Fog(?f) ^ UAV(?u) ^ Radar(?r) ^ cameraInfra(?c) ^ hasSensor(?u, ?c) ^ hasSensor(?u, ?r)
-> isActiveUAV(?u, true)
```

Figure 17. Activation of a UAV.

```
UAV(?u) ^ Radar(?r) ^ cameraInfra(?c) ^ hasSensor(?u, ?c) ^ hasSensor(?u, ?r) ^
isActiveUAV(?u, true) -> isActiveSensor(?r,true) ^ isActiveSensor(?c,true)
```

Figure 18. Activation of the sensors of the UAV.

As previously stated, there are multiple possible communication protocols: a classical RF protocol, a VLC only protocol in case of interferences, and an optimal hybrid protocol with a better quality of service thanks to the redundancy of data. The choice is made according to the environmental situation where the vehicle is evolving in (see Figure 19). VLC relies on light and performs poorly in a bad weather environment because of the diffraction of the raindrops, fog, and snowflakes. Rule 1 covers this situation but only activating the RF protocol in a bad weather environment. However, those characteristics also allow VLC to bring light in spaces with poor illumination, enhancing the general visibility and perception in a dark area. The hybrid approach presents a real interest in a dark environment. This situation is illustrated by Rule 2. In some specific situations, such as in a fire hazard, the illumination of an environment can have an impact on the VLC sensors. Rules 3 and 4 cover two different scenarios where, depending on the brightness of a fire hazard, the VLC protocol is activated or not.

```
1. RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^
hasCommunicationProtocol(?U, ?h) ^ Hybride(?h) ^ VLC(?v) ^
BadWeather(?b) ^ UAV(?U) ^ hasCommunicationProtocol(?U, ?r)
-> isActiveCommunicationProtocol(?r, true) ^
isActiveCommunicationProtocol(?v, false) ^
isActiveCommunicationProtocol(?h, false)

2. RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^ Dark(?d)
^ hasCommunicationProtocol(?U, ?h) ^ Hybride(?h) ^
VLC(?v) ^ UAV(?U) ^ hasCommunicationProtocol(?U, ?r)
-> isActiveCommunicationProtocol(?h, true) ^
isActiveCommunicationProtocol(?v, true) ^
isActiveCommunicationProtocol(?r, true)

3. RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^
FireHazard(?f) ^ hasBrightnessValue(?br, ?bv) ^
hasCommunicationProtocol(?U, ?h) ^
brightnessSensor(?br) ^ Hybride(?h) ^ VLC(?v) ^
UAV(?U) ^ hasCommunicationProtocol(?U, ?r) ^
UnusualEnvironment(?ue) ^ swrlb:greaterThan(?bv, 70)
-> isActiveCommunicationProtocol(?r, true) ^
isActiveCommunicationProtocol(?v, false) ^
isActiveCommunicationProtocol(?h, false)

4. RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^
FireHazard(?f) ^ hasBrightnessValue(?br, ?bv) ^
hasCommunicationProtocol(?U, ?h) ^
brightnessSensor(?br) ^ swrlb:lessThan(?bv, 70) ^
Hybride(?h) ^ VLC(?v) ^ UAV(?U) ^
hasCommunicationProtocol(?U, ?r) ^
UnusualEnvironment(?ue) ->
isActiveCommunicationProtocol(?h, true) ^
isActiveCommunicationProtocol(?v, true) ^
isActiveCommunicationProtocol(?r, true)
```

Figure 19. Rules governing activation of the communication protocols.

4.5.3. Summary of the Process

This work revolves around the perception enhancement of an autonomous vehicle, a process conducted in four steps, as shown in Figure 20:

1. *Environment detection with respect to weather and other hazards*: The system infers what kind of environment the vehicle is advancing in;
2. *Sensor activation*: The system decides what sensors are most appropriate in the current environment;

3. *Entity activation*: The system looks up nearby entities (UAVs) carrying the appropriate sensors;
4. *Communication protocol activation*: The system selects the appropriate communication protocol based on the context of the environment.

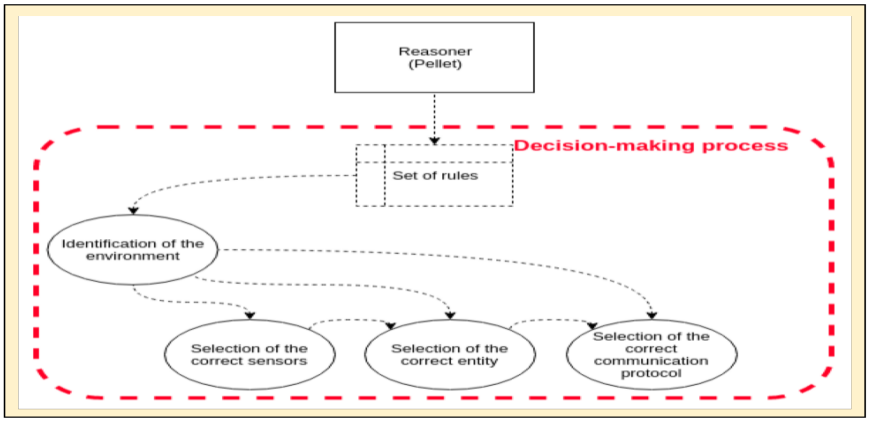

Figure 20. Decision-making process in details.

This modular approach ensures that all stacks of the process are addressed according to their own parameters and context. The environment situation plays the most important role into the proper functioning of these elements. Some other variables can also influence each step, such as if the needed sensors are functional or not, or if the correct ones are present on the UAV or not.

The reasoner consults the set of rules and applies them on the elements classified in the knowledge base. It uses the available elements in order to identify the environment, as shown in Figure 17. This allows the correct identification of the context where the vehicle is evolving and will have a direct impact on succeeding decisions. Thanks to logical rules and data properties, the reasoner can correctly identify which sensors can function well and which ones are weak in a particular situation.

- The knowledge base also contains a list of different entities (UAV) available on site and their embedded sensors, allowing for the system to select which ones should be requested.
- The same knowledge base also includes different communication protocols available on each entity, as well as their respective weaknesses. The logical rules were implemented by taking into consideration all these elements and allowing the reasoner to output different inferred elements: the current environment, requesting assistance on a specific entity, activation on specific sensors, activation of specific communication protocols, and if possible, driving recommendations.

It is to be noted that an inference engine implements all the rules simultaneously, meaning that the results are processed at the same time. However, for a specific situation, if one of the steps is not validated, it will cancel the whole process. For example, if the vehicle is in a rainy area and has no rain sensors, the environment will not be validated, and the perception enhancement process will not happen. If a UAV does not have the correct sensors, then it will not be asked to transmit the perception data.

5. Tests and Validation

As a proof of concept, we describe in this section a use case that will apply all the concepts that were mentioned in the previous sections.

5.1. The Simulator

The simulated environment is a tool that demonstrates the proof of concept of this work. It is via a test scenario that a test subject can browse the environment, be advised of

their next surroundings, and complete the given driving course. Different context elements are initiated here (i.e., vehicles, obstacles, hazards, etc.). Data are generated periodically during the experiment. The driving environment is in constant communication with the knowledge base via a TCP socket interface, sending data of the surroundings to the vehicle and receiving inferred results from the reasoner. The simulator runs in Unity software. More discussions of this simulator are available in the previous paper [4].

The architecture of our simulator is shown in Figure 21. A sample scene in the simulator is shown in Figure 22. The simulated environment is made up of three major components:

- The data generator component manages the virtual data that will be generated and gathered by the system. There are three different types of data:
    - *Driving data*, related to the ego vehicle and controlled by the subject and influenced by the driving environment. All these data are directly influenced by the subject driver. Example: speed, direction of the steering angle, etc.
    - *Environmental data* are data that are captured by the Environment sensors. The simulator allows for the manipulation of different variables, for example, brightness or rain level, and those can be used for the correct identification of the environment.
    - *Perception data* are those gathered by other perception sensors, such as the distance or position of sensors. They are generated by the sensors embedded on the vehicles and UAV, and are used for the identification of other entities on the road.
- The knowledge base interface component manages the interface of the simulated environment and the knowledge base. The generated data are encapsulated in JSON format before being sent to the knowledge base via a TCP socket, and the same channel is used to receive the inference results and driving recommendation.
- The HMI (human–machine interface) component allows for the interaction between the user and the simulator. The actions that the driver takes are shown in the simulator, such as the car moving when the driver uses the accelerator; and the assistance widget displays the results of the inference engine, for example, a warning of the presence of an obstacle ahead.

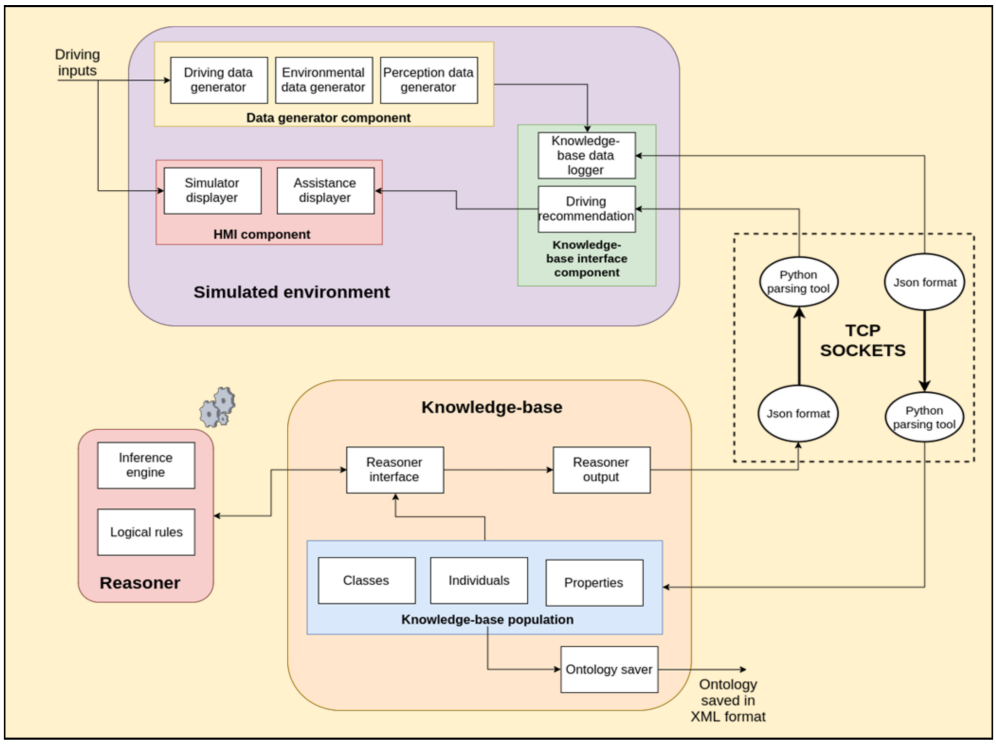

Figure 21. Architecture of the simulator.

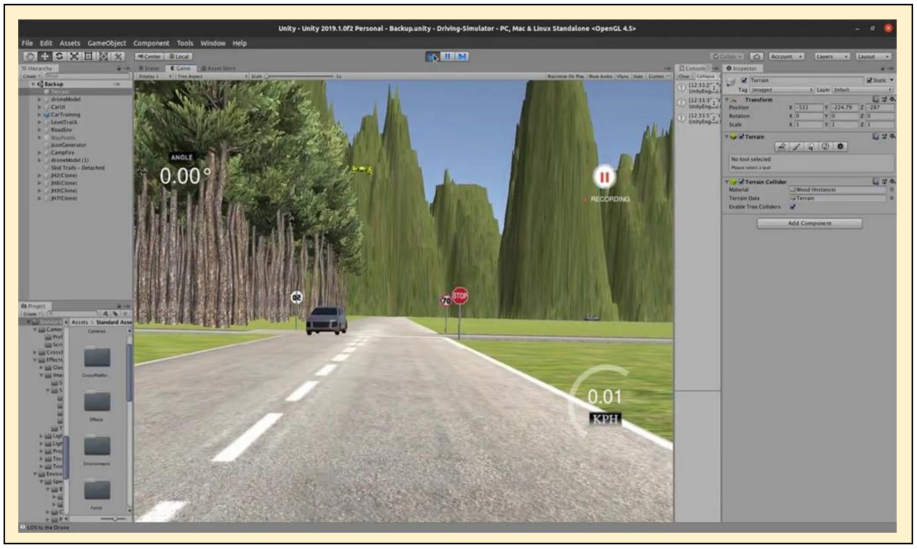

Figure 22. A scene in the simulator (i.e., normal driving).

5.2. Knowledge Base

The knowledge base is where all the data are stored and processed. It comes in the form of an ontology that needs to be setup beforehand. All the necessary classes are initialized; the different individuals of various classes are declared based on the information coming from the simulator and categorized accordingly by class. The ontology is stored in XML format and built thanks to the Protégé software [54].

The data received from the simulator are used in populating the knowledge base: the individuals are stored in the right class, and their properties are also declared. Those elements are sent to the reasoner in order to infer some new conclusions from the existing entities. The reasoner serves as the intelligence layer of the model. The reasoner can infer new information, such as the environment the vehicle is evolving in, or the action that it should take. These deductions are also stored in the knowledge base in specific classes such as Action or Environment. The reasoner used is Pellet, and the logical rules editor is a plug-in to the Protégé software (see Figure 23).

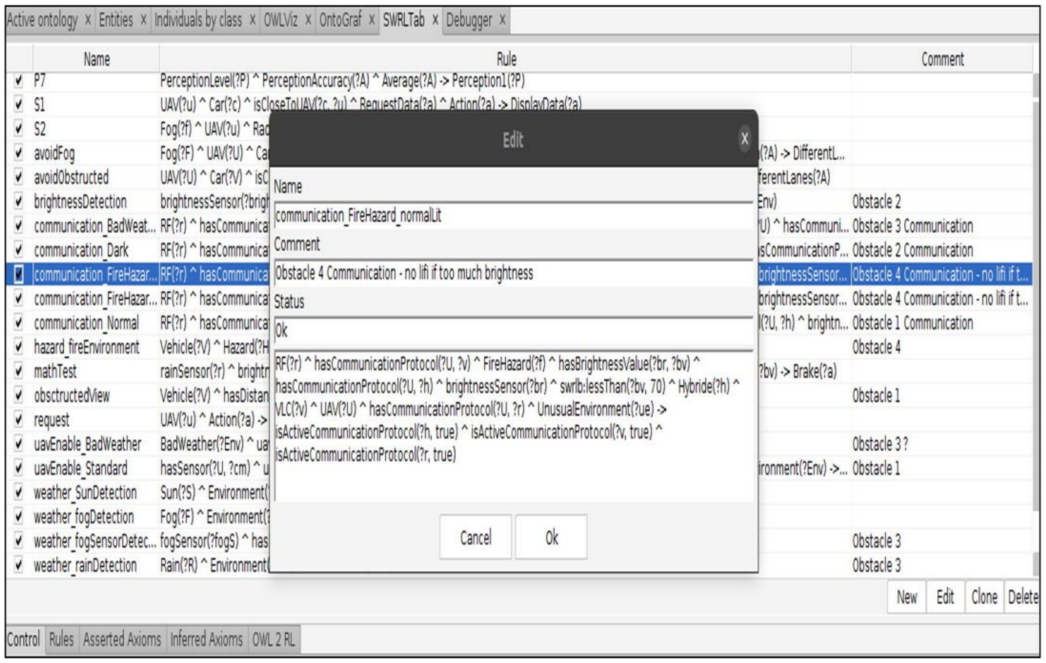

Figure 23. SWRL logical rules editor as it appears in the Protégé software.

5.3. Notification Assistance

The context detection and perception enhancement mechanisms are put in place to assist the driver in mitigating complex driving situations and prevent driving accidents. This is the objective of ADAS (advanced driving assistance systems) [55–57]. This work is related to driving assistance in the sense that if danger is sensed, this is shown on the screen of the HMI component of the simulator. Various notification messages are presented to the driver, such as the ego vehicle's distance to an obstacle. This notification is even more important when the driver's view is hindered by fog or the line-of-sight view is blocked by a topology (e.g., a hill or mountain). This is where the importance of perception enhancement comes into the scene. A drone can see some events happening in a distant position (e.g., fire, blocked traffic, obstacle, etc.) and the driver is notified in advance. As a result, the driver is well-informed of how to mitigate such a danger or obstacle. A sample notification message of the simulator is shown in Figure 24.

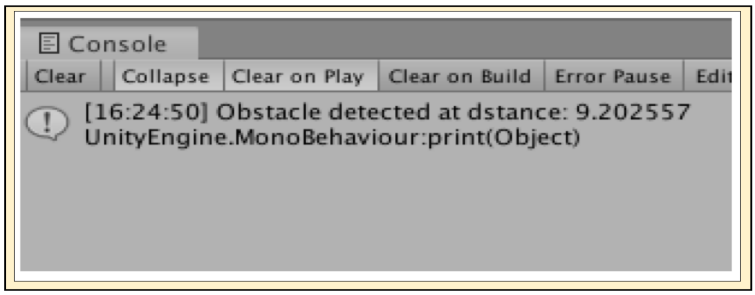

Figure 24. Sample notification assistance from the simulator (obstacle detected).

5.4. Use Case Scenarios

Some 30 subjects were invited to test our simulator and simulate driving in various scenarios. The subjects were a mix of young men and women who had no or very little driving experience. As the scenarios were executed, the subjects' reactions were logged and evaluated. The tested use cases are as follows (see Figure 25):

- The first scenario involves an ego vehicle, going into an intersection with an obstructed view;
- The second scenario involves a rainy environment;
- The third scenario is a foggy environment where visibility is poor. The drone is available to provide added environment perception data;
- The fourth scenario involves driving in a poorly lit environment where the whole area is dark;
- The last scenario is a complex one with an obstructed view, foggy area, and fire hazard present, which the test subject needs to pass.

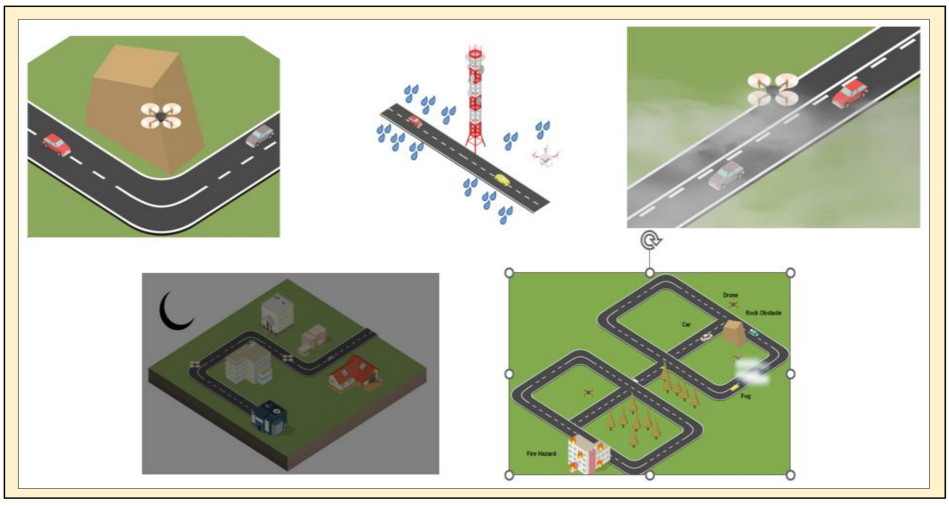

Figure 25. Various use cases were tested to validate our work.

These scenarios cover different situations and validate the concepts presented in this paper. For each of these cases, a test subject's vehicle must complete a circuit with different types of obstacles and events. UAVs are stationed in key points in order to provide further environmental data.

Consider the complex use case shown in Figure 26: the ego vehicle will encounter an obstructed view, a foggy area, and a building on fire. Assume further that multiple UAVs are stationed in key positions. After various sensors take some environment parameters, let us assume that there are some missing data, as shown in Figure 27. Further details of the data-completion process invoked in this work are discussed in [52]. The same data-completion methodology process will be applied here.

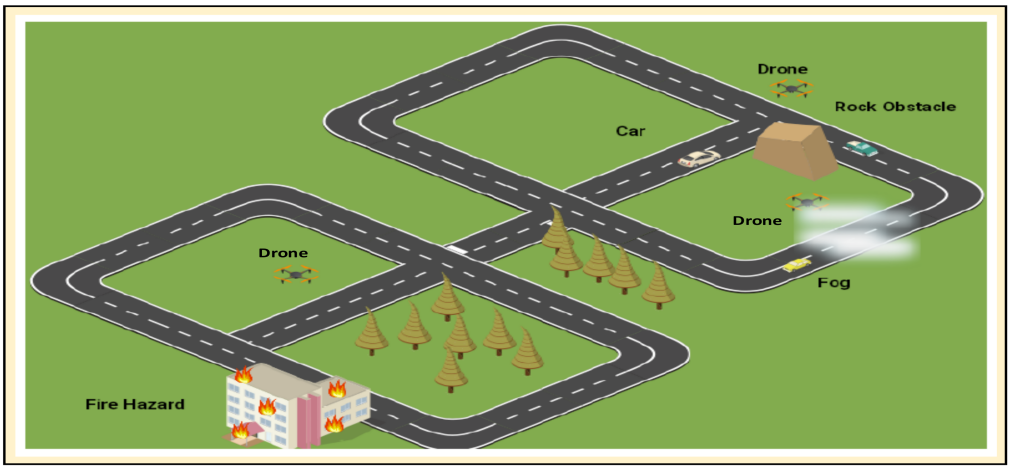

Figure 26. Sample use case scenario involving obstructed view, fog, and fire hazard.

sensorName	embeddedOn	readValue
rainSensor1	mainVehicle	0
brightnessSensor1	mainVehicle	19
rainSensor2	uav1	X
cameraMono1	uav1	400
lidar1	uav1	X
lidar2	mainVehicle	370
radar1	uav2	700

elementName	position	speed
mainVehicle	27;33	59
uav1	X;41	0
uav2	17;45	X
obstacleVehicle1	29;46	30
uav3	53;X	X
obstacleVehicle2	51;39	X

Figure 27. Data obtained with some missing values (represented as x in red).

The data-completion process will complete the missing data in Figure 24. The completed data are now shown in Figure 28. The solution is more than 95% accurate when correctly trained, and thus allows for the *PerceptionAccuracy* value to become *Good*.

Given the complexity of the scenario, multiple sets of rules will be invoked by the reasoner. They can be grouped as per the situation encountered: rules relative to the obstructed view are shown in Figure 29, the ones related to the foggy area are shown in Figure 30, while the set of rules for the fire hazard are presented in Figure 31.

sensorName	embeddedOn	readValue
rainSensor1	mainVehicle	0
brightnessSensor1	mainVehicle	19
rainSensor2	uav1	0
cameraMono1	uav1	400
lidar1	uav1	380
lidar2	mainVehicle	370
radar1	uav2	700

elementName	position	speed
mainVehicle	27;33	59
uav1	27;41	0
uav2	17;45	0
obstacleVehicle1	29;46	30
uav3	53;47	0
obstacleVehicle2	51;39	40

Figure 28. Data set after the completion process (x in red are now replaced by actual values).

```
Vehicle(?V) ^ hasDistanceFromVehicle(?O, NearDistance)
^ Environment(?Env) ^ Object(?O) ^
PerceptionAccuracy(?P)^ Good(?P)^
-> hasObstructedView(?V, ?O)


UAV(?U) ^ cameraStereo(?cs) ^ hasSensor(?U, ?cs) ^
Vehicle(?V) ^ Object(?O) ^ hasObstructedView(?V, ?O) ^
NormalEnvironment(?Env) ^ isCloseToUAV(?V,?U)->
isActiveUAV(?U, true)

UAV(?U) ^ cameraStereo(?cs) ^ hasSensor(?U, ?cs) ^
isActiveUAV(?U,true) -> isActiveSensor(?cs, true)


UAV(?U) ^ RF(?r) ^ VLC(?v) ^ Hybride(?h)
^hasCommunicationProtocol(?U, ?v)
^hasCommunicationProtocol(?U, ?h)
^hasCommunicationProtocol(?U, ?r)
^NormalEnvironment(?E)
^isActiveUAV(?U,true) ->
isActiveCommunicationProtocol(?r, true) ^
isActiveCommunicationProtocol(?v, true) ^
isActiveCommunicationProtocol(?h, true)
```

Figure 29. Set of rules for the obstructed view part of the use case scenario.

```
Vehicle(?V )^ FogSensor(?fogS) ^ hasSensor(?C,?fogS)
^ hasFogValue(?fogS, ?fogV) ^ swrlb:greaterThan(?fogV, 50) ^
PerceptionAccuracy(?P)^ Good(?P)^
^ Weather(?W)
->Fog(?W)

Fog(?F) ^ Environment(?Env)
-> BadWeather(?Env)


Fog(?F) ^ Vehicle(?V) ^ UAV(?u) ^Radar(?ra) ^
cameraInfra(?c)^ hasSensor(?u,?c) ^
hasSensor(?u,?ra) ^ isCloseToUAV(?V,?U)
-> isActiveUAV(?u,true)


UAV(?u) ^ Radar(?ra) ^ cameraInfra(?c)^
hasSensor(?u,?c) ^ hasSensor(?u,?ra) ^ isActiveUAV(?u,true)
-> isActiveSensor(?ra, true) ^ isActiveSensor(?c, true)

UAV(?U) ^ RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^
hasCommunicationProtocol(?U, ?h) ^ Hybride(?h) ^ VLC(?v) ^
BadWeather(?b) ^  hasCommunicationProtocol(?U, ?r)
-> isActiveCommunicationProtocol(?r, true) ^
isActiveCommunicationProtocol(?v, false) ^
isActiveCommunicationProtocol(?h, false)
```

Figure 30. Set of rules for the foggy area part of the use case scenario.

As shown in the logical rules stated in Figure 26, there is a huge rock obstacle near the intersection of two roads, ahead of the ego vehicle. Given that the perception accuracy value in the environment is good, the huge object is considered an obstacle to the vehicle; hence, this case is considered as "*obstructed view*". Next, there is a UAV containing a *stereo camera* in the given situation. Such a UAV is near the ego vehicle; it is the one needed and so this UAV is activated. The *stereo camera* within the UAV is also activated. Finally, the environment is considered normal; hence, all the possible communication protocols (*RF*, *VLC*, and *hybrid*) are activated.

The logical rules related to handling a foggy road situation are shown in Figure 27. As shown, the environment sensor detects a *fog* value greater than the threshold value. As such, it concludes that the concerned area is a *foggy one*, and on a foggy environment, the *BadEnvironment* flag is set to ON. Meanwhile, there is a UAV nearby, near to the ego vehicle. This UAV is equipped with two sensors: *radar* and *infrared camera*, the ones needed for a foggy weather condition. The next two rules activate the UAV as well as the radar and infrared camera. Next, the UAV possesses RF, VLC, and hybrid communication protocols. Given the weather condition, the appropriate logical rule activates the *RF communication protocol*, while deactivating those of the VLC and hybrid.

```
Building(?B) ^ isOnFire(?B, true) ^ Action(?A)^ -> FireHazard(?A)

FireHazard(?A) ^ Environment(?E) ^
PerceptionAccuracy(?P)^ Good(?P)^
-> UnusualEnvironment(?E)


Vehicle(?V) ^ Hazard(?H) ^ Environment(?Env)->UnusualEnvironment(?Env)


UnusualEnvironment(?e) ^ FireHazard(?f) ^ Vehicle(?v) ^
UAV(?u) ^ Radar(?ra) ^ Lidar(?l)^ cameraStereo(?c)^
hasSensor(?u,?c) ^ hasSensor(?u,?l) ^
hasSensor(?u,?ra) ^ isCloseToUAV(?V,?U)
-> isActiveUAV(?u,true)

UAV(?U) ^ Radar(?ra) ^ Lidar(?l)^ cameraStereo(?c)^
hasSensor(?u,?c) ^ hasSensor(?u,?l) ^
hasSensor(?u,?ra) ^ isActiveUAV(?U,true) ->
isActiveSensor(?c, true) ^ isActiveSensor(?l, true)^
isActiveSensor(?ra, true)



RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^ FireHazard(?f)
^ hasBrightnessValue(?br, ?bv) ^
hasCommunicationProtocol(?U, ?h) ^ brightnessSensor(?br) ^
swrlb:lessThan(?bv, 70) ^ Hybride(?h) ^ VLC(?v) ^ UAV(?U)
^ hasCommunicationProtocol(?U, ?r) ^
UnusualEnvironment(?ue) ->
isActiveCommunicationProtocol(?h, true) ^
isActiveCommunicationProtocol(?v, true) ^
isActiveCommunicationProtocol(?r, true)


RF(?r) ^ hasCommunicationProtocol(?U, ?v) ^ FireHazard(?f)
^ hasBrightnessValue(?br, ?bv) ^
hasCommunicationProtocol(?U, ?h) ^ brightnessSensor(?br) ^
Hybride(?h) ^ VLC(?v) ^ UAV(?U) ^
hasCommunicationProtocol(?U, ?r) ^ UnusualEnvironment(?ue)
^ swrlb:greaterThan(?bv, 70) ->
isActiveCommunicationProtocol(?r, true) ^
isActiveCommunicationProtocol(?v, false) ^
isActiveCommunicationProtocol(?h, false)
```

Figure 31. Set of rules for the fire hazard section of the use case scenario.

The logical rules related to handling the fire hazard situation in the use case scenario are shown in Figure 28. First, the given scenario states that there is a building, and it is on fire. The second rule concludes that the area belongs to the *UnusualEnvironment* category. Next, another rule validates that it is an unusual environment. Given the case, it finds a UAV in the environment that has three sensors: *radar*, *lidar*, and *stereo camera*. This UAV is activated. The next rule activates all the three sensors. Due to the ongoing fire, the system gets the *brightness value* in the environment. Two rules decide which communication protocols would be activated. The first one states that if the *brightness value is lesser than*

the threshold value, then the three communication protocols available in the UAV are all activated, namely *RF*, *VLC*, and *hybrid*. The second rule is applicable if *the threshold value of brightness is exceeded*. If so, then only the *RF communication protocol* is activated given that it is the one that is appropriate for the situation. The other two protocols—VLC and hybrid—are deactivated.

6. Results and Discussion

In this proof-of-concept validation, we have split the use case scenario into two subcases. The first subcase is one where sensors and communication protocols are present, while the second subcase is one where some sensors or communication protocols are missing or compromised, hence the system can only offer partial assistance. As expected, when the UAVs with correct sensors are available, coupled with the appropriate communication protocols being present, the system provides full notification assistance. It means that the driver is notified of fog, obstructed view, and fire hazard. When appropriate sensors or communication protocols are missing, the notification assistance is either partial or inexistent (see Figure 32). As a rule of thumb, the driving context recognition is enhanced whenever the appropriate sensors and communication protocols are available. Safe driving is realized because the driver is informed of what lies ahead (i.e., unseen obstacle, fog, fire hazard, etc.).

Event where the UAV is stationed	Obstructed view	Fog	Fire Hazard
Adequate sensors	Yes	Yes	Yes
Adequate communication protocol	Yes	Yes	Yes

Event where the UAV is stationed	Obstructed view	Fog	Fire Hazard
Adequate sensors	Yes	No	Yes
Adequate communication protocol	Yes	Yes	No

Figure 32. Full driver notification assistance (i.e., sensors and communication protocols present) vs. partial notification assistance (i.e., sensors or communication protocol missing). Green color denotes ideal situation where full assistance is achieved. Red color denotes partial or no driving assistance due to absence of correct sensor or correct communication protocol.

In addition to using UAVs as extra tools to enhance driving context perception, this work's contribution lies in offering driving assistance in the form of notification. Early notification of what lies ahead (e.g., obstacle, danger, etc.) allows the driver to plan ahead on how to get over the obstacle or danger that lies ahead within his path. This reduces the possibility of road accidents. Early driving notification is one component of ADAS (advanced driving assistance systems) [57]. It was shown on many occasions that with ADAS, accident prevention is enhanced.

As stated earlier, some test subjects were invited to run the use case scenarios. Three types of driving assistance were conceived: (i) *non-assisted drivers*, (ii) *partially assisted drivers*, and (iii) *fully assisted drivers* (see Figure 33). In the first case, a driver is left on their own. There are no UAVs that seek extra information from the environment, and as a result, there is no driving notification assistance provided to the driver. The result is that the vehicle's speed is a little bit slower. The time to complete the circuit is also longer. Moreover, two incidents (i.e., accident or infraction) were reported. In the second case, the assistance is partial due to an absence of the correct sensor/s within the UAVs or the absence of appropriate communication protocol that would transfer the data from the UAV to the vehicle. There is an improvement here in the sense that drivers were able to complete the driving circuit in faster time. Finally, in a fully assisted driving scenario, the UAVs are equipped with the right sensors and the suitable communication protocols are available.

As a result, the advanced driving notification assistance scheme is fully intact. The test results indicate that drivers drive (i.e., average speed and maximum speed) faster, and the time to complete the circuit is a lot faster. It is to be noted, however, that an incident may still happen even with driving assistance notification. This only means that human error is always present, even though the quantity of such human errors is somewhat reduced.

	Average speed	Time to complete the experiment	Max speed	Incident
Non-assisted driver 1	57km/h	118s	126km/h	0
Non-assisted driver 2	37km/h	161s	69km/h	2
Partially assisted driver 1	55km/h	108s	97km/h	0
Partially assisted driver 2	49km/h	121s	80km/h	1

	Average speed	Time to complete the experiment	Max speed	Incident
Fully assisted driver 1	73km/h	96s	119km/h	0
Fully assisted driver 2	90km/h	89s	130km/h	1
Fully assisted driver 3	79km/h	99s	111km/h	0

Figure 33. Results of the driving test simulation among test subjects.

The generalizations that can be deduced from the results of the use case simulations are the following:

- Drivers are fully assisted when UAVs with both appropriate sensors and communication protocols are present. The assistance is rendered in the form of advanced driving notification (e.g., obstacle or danger that lies ahead).
- Drivers are only partially assisted when either the appropriate sensors or communication protocols are missing. Drivers are not assisted and on their own whenever both appropriate sensors and communication protocols are absent in the UAVs; hence, the UAVs cannot be used as extra tools to enhance driving context perception.
- When drivers are fully assisted, the time for the pilot to complete the test circuit in the experiment is faster. Their average and maximum vehicular speeds improve significantly.
- Drivers that are not or only partially assisted finished the test circuit longer. Their speeds are slower and incidents are higher.
- The subjects drive faster when they already know some events coming ahead of time due to full notification assistance.
- There is minimal incident among the drivers when they have full notification assistance.

7. Conclusions and Future Work

In this paper, we have shown that a UAV can be used as an extra layer for environmental perception. In the case of an autonomous vehicle, given that the safety of road users and vehicle passengers is paramount, then the perception of the driving environment must be correct for the ego vehicle to react accordingly to a given driving situation. Apart from the usual sensors in the vehicle and in the smart city, a UAV can also be equipped with necessary sensors to perceive its environment. The UAV can be deployed to gather environmental data, which will then be sent to the knowledge base for fusion with other data. The result is a richer collection of data related to the perception of the driving environment. The knowledge base is represented by ontology; it contains various logical rules related to the weather, the darkness of the environment, the appropriateness of sensors with respect to the weather and darkness, and the activation mechanism of UAVs containing these sensors. Logical rules about which communication protocols to use also exist. Finally, various driving context cognition rules and assistance are provided. Overall, the use of UAVs supplies additional information that enables an autonomous vehicle to enhance its environment perception. As a result, the ego vehicle reacts appropriately (even knowing

what lies ahead) to the driving situation, further enhancing driving safety and reducing road accidents. Various use cases are used as proofs of concept and validate our concept.

The contributions of this work are as follows:

- A new application of UAVs, that is, to equip them with necessary sensors and communication protocols so that they can be used to supply external perception to an autonomous vehicle;
- Knowledge representation using ontology and a functional knowledge base, and a set of logical rules for context detection and perception enhancement;
- A hybrid VLC/RF communication protocol for safer transmission;
- A driving simulator with a realistic physics engine for testing scenarios.

Even though this work is original in many respects and functional in its own right, as a future work, there are still various ideas around making the system work even better. For instance, it is possible to improve logical rules by considering additional classes of objects found on the road [58]. It might be possible to use machine learning [59–61] for autogeneration of logical rules and to improve the hybrid communication protocol hardware and software.

Author Contributions: Conceptualization, A.K., M.D.H. and A.R.-C.; methodology, A.K. and A.R.-C.; software, A.K.; validation, A.K., M.D.H. and A.R.-C.; formal analysis, A.K., M.D.H. and A.R.-C.; writing—original draft preparation, A.K. and M.D.H.; writing—review and editing, M.D.H.; supervision, M.D.H. and A.R.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by ECE Paris Engineering School, and LISV Laboratory, Université de Versailles—Paris-Saclay.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wikipedia. List of Countries by Vehicles per Capita. Available online: https://en.wikipedia.org/ (accessed on 1 June 2022).
2. Fayyad, J.; Jaradat, M.A.; Gruyer, D.; Najjaran, H. Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review. *Sensors* **2020**, *20*, 4220. [CrossRef] [PubMed]
3. Zheng, R.; Liu, C.; Guo, Q. A decision-making method for autonomous vehicles based on simulation and reinforcement learning. In Proceedings of the International Conference on Machine Learning and Cybernetics, Tianjin, China, 14–17 July 2013.
4. Khezaz, A.; Hina, M.D.; Guan, H.; Ramdane-Cherif, A. Knowledge-Based Approach for the Perception Enhancement of a Vehicle. *J. Sens. Actuator Netw.* **2021**, *10*, 66. [CrossRef]
5. Das, S.; Brimley, B.K.; Lindheimer, T.E.; Zupancich, M. Association of reduced visibility with crash outcomes. *IATSS Res.* **2018**, *42*, 143–151. [CrossRef]
6. WHO (World Health Organization). 10 Facts on Global Road Safety. Available online: http://www.who.int/features/factfiles/roadsafety/en/ (accessed on 20 January 2022).
7. WHO (World Health Organization). Save LIVES: A Road Safety Technical Package. 2017. Available online: https://www.who.int/publications/i/item/save-lives-a-road-safety-technical-package (accessed on 1 June 2022).
8. Van Brummelen, J.; O'Brien, M.; Gruyer, D.; Najjaran, H. Autonomous vehicle perception: The technology of today and tomorrow. *Transp. Res. Part C Emerg. Technol.* **2018**, *89*, 384–406. [CrossRef]
9. Campbell, M.; Egerstedt, M.; How, J.P.; Murray, R.M. Autonomous driving in urban environments: Approaches, lessons and challenge. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2010**, *368*, 4649–4672. [CrossRef]
10. Vanholme, B.; Gruyer, D.; Lusetti, B.; Glaser, S.; Mammar, S. Highly Automated Driving on Highways Based on Legal Safety. *IEEE Trans. Intell. Transp. Syst.* **2012**, *14*, 333–347. [CrossRef]
11. Zyda, M. Creating a Science of Games: Introduction. *Commun. ACM Spec. Issue Creat. A Sci. Games* **2007**, *50*, 26–29.
12. Mayhan, R.J.; Bishel, R.A. A two-frequency radar for vehicle automatic lateral control. *IEEE Trans. Veh. Technol.* **1982**, *31*, 32–39. [CrossRef]

13. Du, Y.; Man, K.L.; Lim, E.G. Image Radar-based Traffic Surveillance System: An all-weather sensor as intelligent transportation infrastructure component. In Proceedings of the 2020 International SoC Design Conference (ISOCC), Yeosu, Korea, 21–24 October 2020.
14. Rasshofer, R.H.; Gresser, K. Automotive Radar and Lidar Systems for Next Generation Driver Assistance Functions. *Adv. Radio Sci.* **2005**, *3*, 205–209. [CrossRef]
15. Guerrero-Ibáñez, J.; Zeadally, S.; Contreras-Castillo, J. Sensor Technologies for Intelligent Transportation Systems. *Sensors* **2018**, *18*, 1212. [CrossRef]
16. Sivaraman, S.; Trivedi, M.M. Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. *IEEE Trans. Intell. Transp. Syst.* **2013**, *14*, 1773–1795. [CrossRef]
17. Sahin, F. Long-Range, High-Resolution Camera Optical Design for Assisted and Autonomous Driving. *Photonics* **2019**, *6*, 73. [CrossRef]
18. Smith, M. *Light Detection and Ranging (LIDAR)*; Volume 2, A Bibliography with Abstracts, Progress Report, 1975; National Technical Information Service: Springfield, VA, USA, 1978.
19. Li, B.; Zhang, T.; Xia, T. Vehicle Detection from 3D Lidar Using Fully Convolutional Network. *arXiv* **2016**, arXiv:1608.07916.
20. Mahlisch, M.; Schweiger, R.; Ritter, W.; Dietmayer, K. Sensor fusion Using Spatio-Temporal Aligned Video and Lidar for Improved Vehicle Detection. In Proceedings of the IEEE Intelligent Vehicles Symposium, Meguro-Ku, Japan, 13–15 June 2006.
21. Ghazzai, H.; Menouar, H.; Kadri, A. On the Placement of UAV Docking Stations for Future Intelligent Transportation Systems. In Proceedings of the IEEE 85th Vehicular Technology Conference, Sydney, NSW, Australia, 4–7 June 2017.
22. Starship Technologies. Starship. Available online: https://www.starship.xyz (accessed on 15 January 2022).
23. Google. Google X Wing. Available online: https://wing.com/ (accessed on 15 January 2022).
24. Amazon. Amazon Prime Air. Available online: www.amazon.com/primeair (accessed on 12 March 2022).
25. Sakiyama, M. The Balance between Privacy and Safety in Police UAV Use: The Power of Treat and Its Effect on People's Receptivity. Ph.D. Thesis, University of Nevada, Reno, NV, USA, 2017.
26. Xu, Y.; Yu, G.; Wang, Y.; Wu, X.; Ma, Y. Car Detection from Low-Altitude UAV Imagery with the Faster R-CNN. *J. Adv. Transp.* **2017**, *2017*, 1–10. [CrossRef]
27. Hadiwardoyo, S.A.; Hernández-Orallo, E.; Calafate, C.T.; Cano, J.C.; Manzoni, P. Experimental characterization of UAV-to-car communications. *Comput. Netw.* **2018**, *136*, 105–118. [CrossRef]
28. Menouar, H.; Guvenc, I.; Akkaya, K.; Uluagac, A.S.; Kadri, A.; Tuncer, A. UAV-Enabled Intelligent Transportation Systems for the Smart City: Applications and Challenges. *IEEE Commun. Mag.* **2017**, *55*, 22–28. [CrossRef]
29. Shi, W.E.A. Drone Assisted Vehicular Networks: Architecture, Challenges and Opportunities. *IEEE Netw.* **2018**, *32*, 130–137. [CrossRef]
30. Galkin, B.; Kibilda, J.; DaSilva, L.A. UAVs as Mobile Infrastructure: Addressing Battery Lifetime. *IEEE Commun. Mag.* **2019**, *57*, 132–137. [CrossRef]
31. Grimm, S.; Hitzler, P.; Abecker, A. Knowledge Representation and Ontologies. In *Semantic Web Services: Concepts, Technology and Applications*; Studer, R., Grimm, S., Abecker, A., Eds.; Springer: Berlin, Germany, 2007; pp. 51–106.
32. Guarino, N. Formal ontology, conceptual analysis and knowledge representation. *Int. J. Hum.-Comput. Stud.* **1995**, *43*, 625–640. [CrossRef]
33. O'Keefe, R.M.; Balci, O.; Smith, E.P. *Validation of Expert System Performance*; Virginia Polytechnic Institute & State University: Blacksburg, VA, USA, 1987; Volume 2.
34. Jaffar, J.; Maher, M.J. Constraint logic programming: A survey. *J. Log. Program.* **1994**, *19–20*, 503–581. [CrossRef]
35. Paulheim, H. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semant. Web* **2016**, *8*, 489–508. [CrossRef]
36. Noy, N.F.; McGuinness, D.L. Ontology Development 101: A Guide to Creating Your First Ontology. Available online: http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html (accessed on 5 July 2017).
37. Horridge, M.; Bechhofer, S. The OWL API: A Java API for OWL ontologies. *Semant. Web* **2011**, *2*, 11–21. [CrossRef]
38. Regele, R. Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles. In Proceedings of the 4th International Conference on Autonomic and Autonomous Systems (ICAS'08), Gosier, France, 16–21 March 2008.
39. Maalel, A.; Mabrouk, H.H.; Mejri, L.; Ghezela, H.H.B. Development of an ontology to assist the modeling of accident scenario application on railroad transport. *J. Comput.* **2011**, *3*, 1–7.
40. Hulsen, M.; Zollner, J.M.; Weiss, C. Traffic Intersection Situation Description Ontology for Advanced Driver Assistance. In Proceedings of the IEEE Intelligent Vehicles Symposium, 2011 IEEE Intelligent Vehicles Symposium, Baden, Germany, 5–9 June 2011.
41. Lahat, D.; Adali, T.; Jutten, C. Multimodal data fusion: An overview of methods, challenges, and prospects. *Proc. IEEE* **2015**, *103*, 1449–1477. [CrossRef]
42. Llinas, J.; Hall, D. An introduction to multisensor data fusion. *Proc. IEEE* **1997**, *85*, 6–23.
43. Hina, M.D.; Thierry, C.; Soukane, A.; Ramdane-Cherif, A. Cognition of Driving Context for Driving Assistance. *Int. J. Comput. Inf. Eng.* **2018**, *12*, 56–66.
44. Bobzien, S. (Ed.) *The Stanford Encyclopedia of Philosophy Metaphysics Research Lab*; Stanford University: Stanford, CA, USA, 2020.
45. W3C. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Available online: https://www.w3.org/Submission/SWRL/ (accessed on 15 January 2022).

46. O'Connor, M.; ShankarMark, R.V.; Musen, M.A.; Das, A.; Nyulas, C. The SWRLAPI: A Development Environment for Working with SWRL Rules. In Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, Karlsruhe, Germany, 26–27 October 2008.
47. Liu, C.-H.; Chang, K.-L.; Chen, J.; Hung, S.-C. Ontology-Based Context Representation and Reasoning Using OWL and SWRL. In Proceedings of the 8th Annual Communication Networks and Services Research Conference, Montreal, QC, Canada, 11–14 May 2010.
48. Mir, A.; Hassan, A. Fuzzy Inference Rule-based Neural Traffic Light Controller. In Proceedings of the IEEE International Conference on Mechatronics and Automation, Changchun, China, 5–8 August 2018.
49. Fernandez, S.; Ito, T. Driver Classification for Intelligent Transportation Systems Using Fuzzy Logic. In Proceedings of the ITSC 2016, IEEE International Conference on Intelligent Transportation Systems, Rio de Janeiro, Brazil, 1–4 November 2016.
50. Azim, T.; Jaffar, M.A.; Mirza, A.M. Fully automated real time fatigue detection of drivers through fuzzy expert systems. *Appl. Soft Comput.* **2014**, *18*, 25–38. [CrossRef]
51. Zadeh, L.A. Is there a need for fuzzy logic? *Inf. Sci.* **2008**, *178*, 2751–2779. [CrossRef]
52. Khezaz, A.; Hina, M.D.; Guan, H.; Ramdane-Cherif, A. Hybrid Machine Learning Model for Traffic Forecasting. *Lect. Notes Inst. Comput. Sci. Soc. Inform. Telecommun. Eng.* **2021**, *372*, 188–199.
53. Chen, V.C.; Li, F.; Ho, S.-S.; Wechsler, H. Micro-Doppler effect in radar: Phenomenon, model, and simulation study. *IEEE Trans. Aerosp. Electron. Syst.* **2006**, *42*, 2–21. [CrossRef]
54. Protégé. Protégé: Open-Source Ontology Editor. Available online: http://protege.stanford.edu (accessed on 15 June 2021).
55. Thalen, J.P. *ADAS for the Car of the Future*; University of Twente: Enschede, The Netherlands, 2006.
56. Li, L.; Wen, D.; Zheng, N.-N.; Shen, L.-C. Cognitive Cars: A New Frontier for ADAS Research. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 395–407. [CrossRef]
57. Hina, M.D.; Guan, H.; Soukane, A.; Ramdane-Cherif, A. CASA: An Alternative Smartphone-Based ADAS. *Int. J. Inf. Technol. Decis. Mak.* **2021**, *21*, 273–313. [CrossRef]
58. Shinzato, P.Y.; Wolf, D.F.; Stiller, C. Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion. In Proceedings of the 2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, USA, 8–11 June 2014; pp. 687–692.
59. Ithape, A.A. Artificial Intelligence and Machine Learning in ADAS. In Proceedings of the Vector India Conference, Pune, India, 18–19 July 2017.
60. Hina, M.D.; Thierry, C.; Soukane, A.; Ramdane-Cherif, A. Ontological and Machine Learning Approaches for Managing Driving Context in Intelligent Transportation. In Proceedings of the KEOD 2017, 9th International Conference on Knowledge Engineering and Ontology Development, Madeira, Portugal, 1–3 November 2017.
61. Jahangiri, A.; Rakha, H.A. Applying Machine Learning Techniques to Transportation Mode Recognition Using Mobile Phone Sensor Data. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2406–2417. [CrossRef]