World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:12, No:2, 2018

# Cognition of Driving Context for Driving Assistance

Manolo Dulva Hina, Clement Thierry, Assia Soukane, Amar Ramdane-Cherif

***Abstract*—**In this paper, we presented our innovative way of determining the driving context for a driving assistance system. We invoke the fusion of all parameters that describe the context of the environment, the vehicle and the driver to obtain the driving context. We created a training set that stores driving situation patterns and from which the system consults to determine the driving situation. A machine-learning algorithm predicts the driving situation. The driving situation is an input to the fission process that yields the action that must be implemented when the driver needs to be informed or assisted from the given the driving situation. The action may be directed towards the driver, the vehicle or both. This is an ongoing work whose goal is to offer an alternative driving assistance system for safe driving, green driving and comfortable driving. Here, ontologies are used for knowledge representation.

***Keywords*—**Cognitive driving, intelligent transportation system, multimodal system, ontology, machine learning.

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

## I. Introduction

THE cognition of driving situation is essential in a driving assistance system. In our work, the driving situation or driving context is the combined contexts of the environment, the vehicle and the driver. The contexts of the environment, vehicle and the driver are themselves the combined parameters that indicate the state of the concerned entity. Ontology is used as a tool to represent knowledge, in this case the one related to context. Ontology is an accepted tool/standard in computing medialization.

### *A. Knowledge Representation Using Ontology*

In computer science, ontology is a formal naming and definition of the types, properties, and interrelationships of the entities that fundamentally exist for a particular domain of discourse. Ontology compartmentalizes the variables needed for some set of computations and establishes the relationships between them. In the fields of artificial intelligence, semantic web and systems engineering ontologies are used to limit complexity and to organize information. The ontology can then be applied to problem solving [1]. There are several definitions or meanings attributed to this concept. The definitions the most common are those of [2], [3]. The first definition of the ontology [2] was: "the terms and the basic relationship with the vocabulary of a domain as well as the rules for combining terms and relations in order to define extensions of such vocabulary". Reference [4] proposed the definition most cited: "an explicit specification of a conceptualization". Reference [3] refines the definition by considering ontologies as partial and formal specifications of a conceptualization. Ontologies are formal because they are expressed as formalism with formal semantics. They are partial because a conceptualization cannot always be fully formalized in such a framework. Ontologies were used in modeling of transport accident [5] and on assistance on search of data [6]. Reference [7] uses ontology in modeling an intelligent driver assistance system (I-DAS) for vehicle safety. There are four types of components to formalize knowledge embedded in ontology, namely: the classes, the relations, the axioms and the instances.

- The *classes of concepts* are a set of words representing an abstract idea or a class of tangible objects.
- The *relation* represents a type of interaction between the concepts of a domain.
- The *axioms* are for structuring of sentences that are always true.
- The *instances* are used to represent the elements.

In order to design our ontology model, we use the software tool called Protégé [8]. We also use VOWL (Visual Notation for OWL) [9] as a plug-in for data visualization. All diagrams related to ontology that appear in this paper are data visualizations through VOWL.

### *B. Driving Context of an Intelligent Transportation*

Our vision of an intelligent vehicle [10], [11] is shown in Fig. 1. An intelligent vehicle is able to sense its driving context, and provides comfort to its user. It is able to communicate with other vehicle (V2V) to navigate safe passage for both vehicles. It is capable of communicating with available infrastructure to make sense of its environment and services that are available. It is able to communicate with other entities such as V2H (vehicle to home) and other interconnected objects, otherwise known as the Internet of Things (IoT) [12].

## II. Related Works

This work is a continuation of our previous work [13], [14]. In our work, the driving context is the result of the fusion of various parameters that describe the environment context, the vehicle context and the driver context. We have compared our definition of driving context with other related works, such as [15], [16]. Our work is evolving; we wish to consider implementing the following changes. We intend to add the relation between the entities as introduced in [17] to add a range of action for a more comfortable driving. We will configure communication and V2I, introduced in [18], as a global context vision instead of agent-centric one. It is a good

M. D. Hina and A. Soukane are associate professors in Computer Science at ECE Paris School of Engineering, 37 quai de Grenelle, 75015 Paris, France (e-mail: manolo-dulva.hina@ece.fr, assa.soukane@ece.fr).

C. Thierry is a master's student in Computer Science and is doing training at ECE Paris School of Engineering, 37 quai de Grenelle, 75015 Paris, France (e-mail: clement.thierry@ece.fr).

A. Ramdane-Cherif is a full professor at LISV Laboratory, University of Versailles St-Quentin-en-Yvelines, 10-12 avenue de l'Europe, 78140 Velizy, France (e-mail: rca@lisv.uvsq.fr).

idea to consider the case of not strictly following traffic rules when a situation demands for it, as in [19]. Finally in [19], the concept of "values" may be warranted to add into our context rules for comfort, green and secured driving.

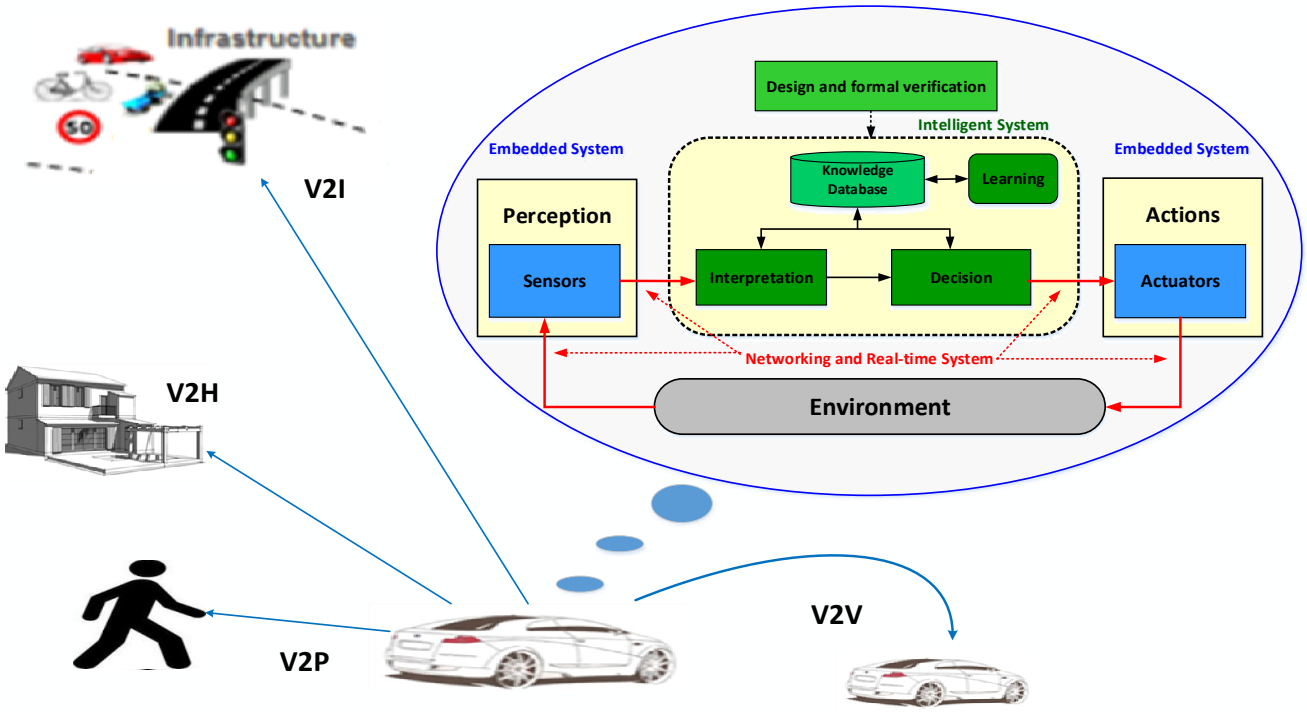



Fig. 1 The intelligent vehicle of tomorrow.

## III. The Driving Model

The driving model [20], [21] is the representation of driving situations and the rules of driving. A driving model is important because through it that we will be able to analyze the driving event and the kind of driving assistance that is appropriate for such event. We will represent the driving model using ontology.

### A. The Driving Context

For the driving context [22], we are concerned about the fusion of three main contexts: that of the vehicle, the driver and the environment. The "*Environment*" is the ontology class that describes the external environment where the human-vehicle interaction exists. The "*Vehicle*" is the class that represents a vehicle used to interact with its driver while the "*Driver*" is the class describing the driver of a vehicle. See Fig. 2.

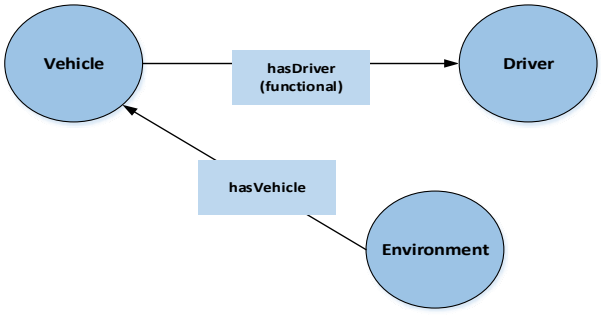



Fig. 2 Ontology for the driving context

### B. The Context of the Driver

The ontological context of the driver is shown in Fig. 3. As shown, the "*Driver*" class is related to other classes, which describe the state of the driver:

- *MentalState*: the current mental state of the driver. The class has the following subclasses: "*Anger*", "*Stress*", "*Fatigue*" and "*Faint*".
- *DriverProfile*: this class contains pertinent attributes: *Name*, *LicenseScore*, *DriverLicenseNumber*, *Age*, etc.
- *DriverViolations*: this class contains the historical driving information. It has subclasses, such as "*RedLightViolation*", "*OverSpeeding*", etc.
- *FocusOnDriving*: a class that contains many Boolean attributes, including "*hasEyesOnTheRoad*", "*hasPhoneConversation*", "*hasPassengerOnBoard*", "*hasHandsOnTheSteeringWheel*", etc.

### C. The Context of the Vehicle

The ontological context of the vehicle is shown in Fig. 4. The class "*Vehicle*" is a subclass of "*MovingObject*" which is a class that describes all moving entities on the road, including pedestrians, cyclists, and other vehicles. A "*Vehicle*" has subclasses, namely "*Truck/Bus*", "*Car*", and "*MotorBike*". The "*Vehicle*" can be described through its relationships with various other classes, given below:

- *TechnicalData*: this class describes the technical data of our referenced vehicle. Its subclasses are "*FuelType*", "*EmissionClass*" and "*TractionType*".
- *Cockpit*: a class that contains the status of all elements

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

that are found in a vehicle's cockpit. For example, '*hasWindowsOpen*' is a data property that has a Boolean value.

- *ComponentsStatus*: this contains as subclasses all components that we have to check to guarantee a good driving experience. Among these subclasses are "*DirectionIndicator*" (with values '*NoIndicator*', '*RightIndicator*', '*LeftIndicator*', and '*DoubleIndicators*'), "*TyresPression*", "*LubricantTemperature*", "*EngineLubricantLevel*" (with values '*LowLevel*', '*HalfLevel*' and '*FullLevel*'), and "*FuelQuantity*". The class has also some Boolean properties to check if some components are active or not. Example is '*hasFogLightsOn*'.
- *hasPossibleCollision*: this is a property that associates our vehicle with the class "*MovingObjects*"
- *hasPhysics*: this property links our vehicle with class "*Physics*" which describes our vehicle with attributes, such as speed and acceleration.

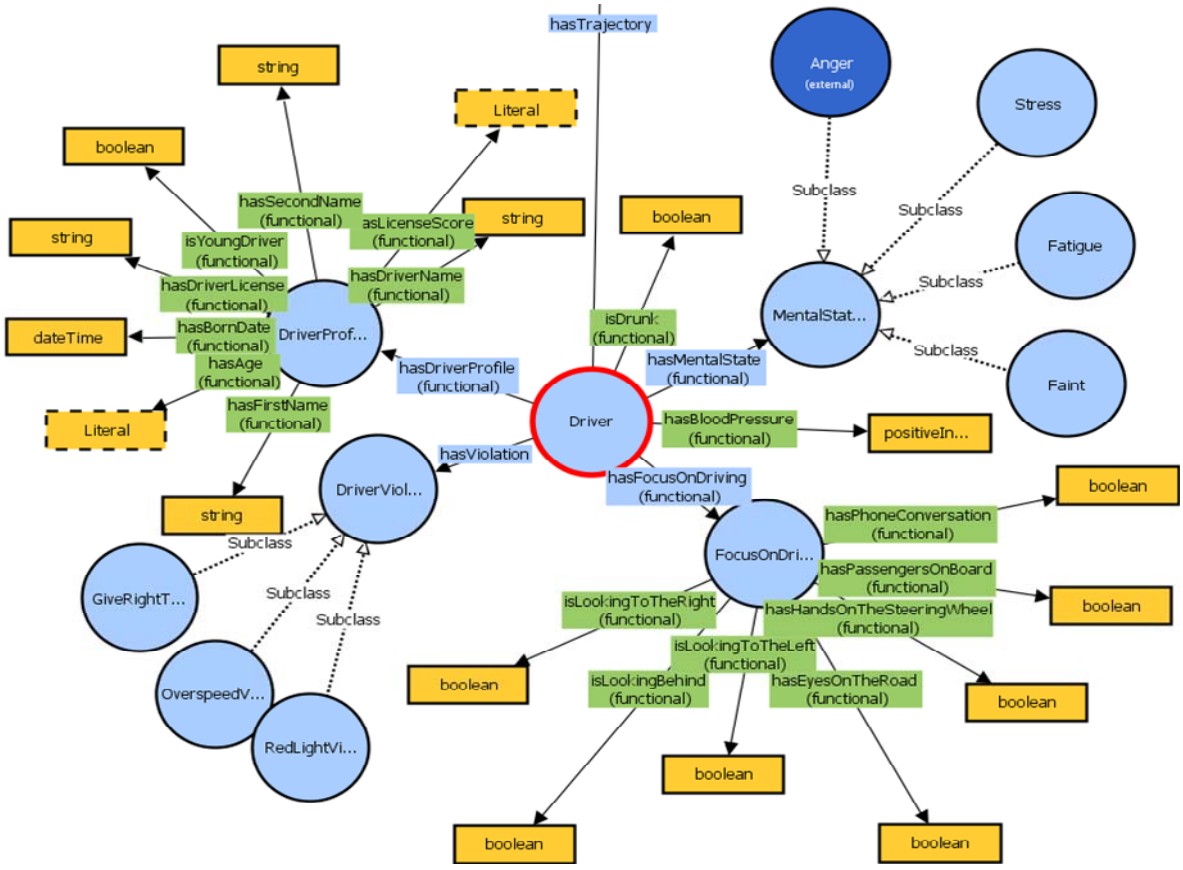



Fig. 3 The ontological representation of the driver's context

### *D. The Context of the Environment*

As shown in Fig. 5, the Environment [23]-[25] is related to all the elements that belong to the scene where the driving event takes place. The "*Environment*" is an abstract class and general concept made up of cities where vehicles are present. The class Environment is related to other classes given below:

- *City*: In our work, an Environment is an area where we can find many cities. A city has two data properties, namely '*hasCityName*' and '*hasLimitedTrafficZone*' which is a Boolean value indicating if the city can be accessed only during some intervals of the day.
- *DistrictArea*: it contains as objects the different districts of a city, linked through '*hasDistrictArea*' property. The position of the "*Driver*" is stored in the "*PositionArea*", a subclass of "*Physics*" and equivalent to "*DistrictArea*".
- *Road*: a road has many data properties, such as '*hasMinSpeedLimit*', '*hasMaxSpeedLimit*', '*hasNumberOfLanes*', '*hasContinuousLine*' and '*hasLength*'. A road is made up of three subclasses, as follows: (1) '*Urban*', (2) '*ExtraUrban*', and (3) '*Highway*'. The *Road* is related to the class *RoadSegment* via property *hasRoadSegment*. A *RoadSegment* has three subclasses: *Lane*, *Intersection*, and *RoundAbout*. They describe the kind of road segments that we find in real life.
- *RoadProperty*: it further describes the "*Road*". Its subclasses include "*Visibility*" (values are '*Low*', '*Average*' or '*High*'), "*Weather*" (values are '*Fog*', '*Sun*', '*Rain*' and '*Snow*'), "*AccidentHistory*" (values are '*Unusual*' or '*Frequent*'), "*TrafficCongestionHistory*" (values are '*Low*', '*Average*' or '*Intense*') and "*CurrentTrafficCongestion*" (values are '*Low*', '*Average*'

or '*Intense*').

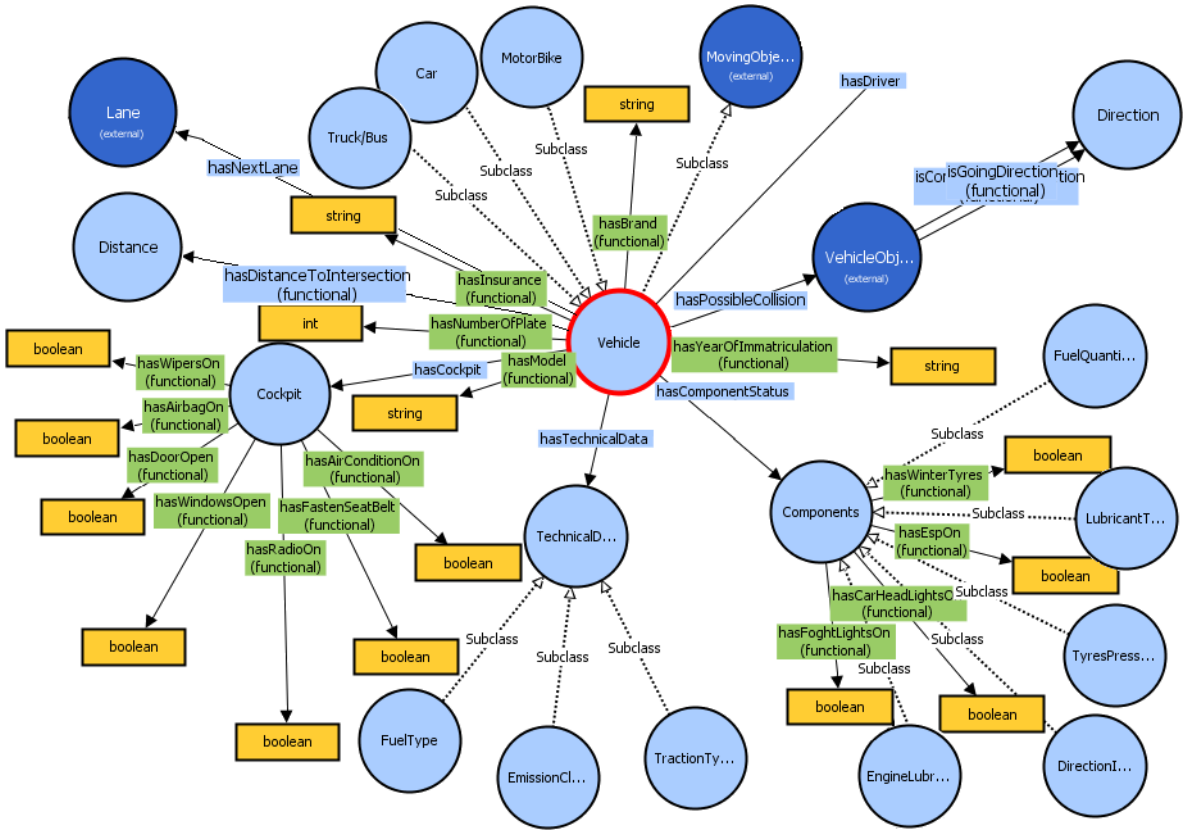



Fig. 4 The ontological representation of the context of the vehicle

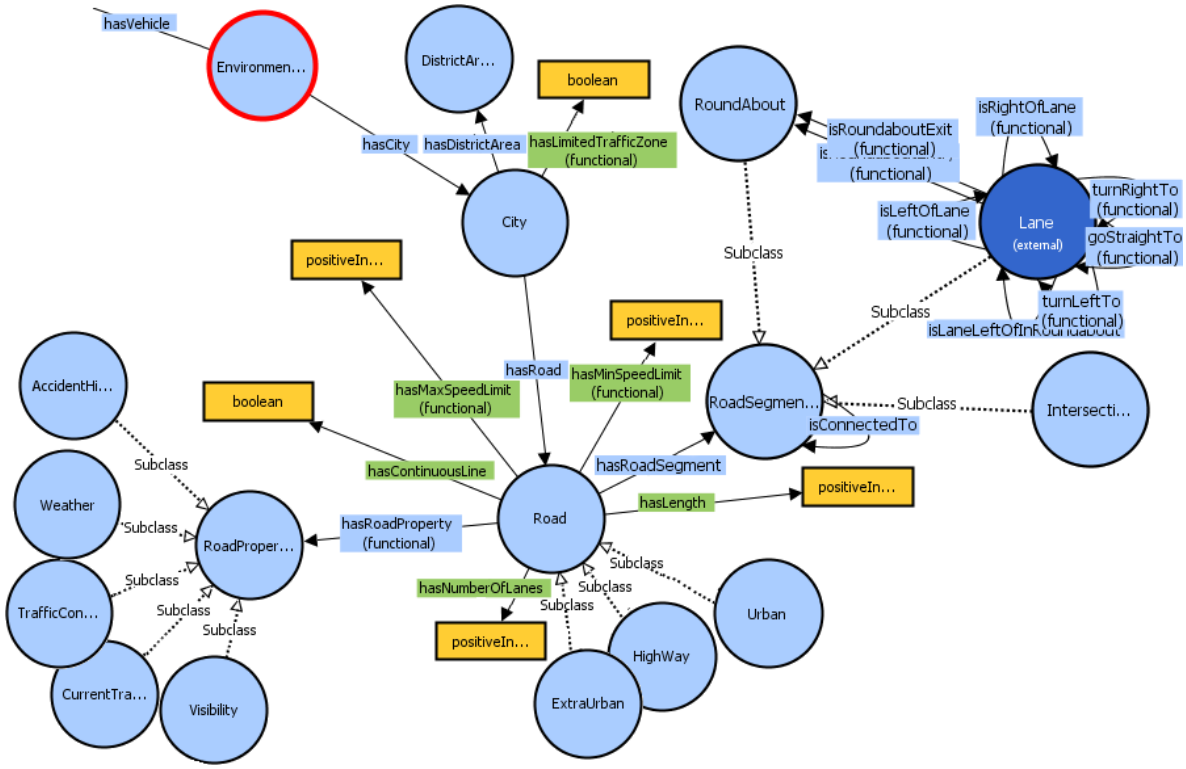



Fig. 5 The ontological representation of the context of the environment

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

### E. Physical Parameters and Trajectory

In this work, all physical parameters that are useful in our ontology are made of subclasses of the general class "*Physics*", which is a class that groups all physical properties that an object can have. The "*Trajectory*" class describes the trajectory of our moving vehicle. A trajectory has start time and has a starting position. Every object is connected with its corresponding "*Physics*" individual via '*hasPhysics*' property. Hence, an object can have multiple properties, such as:

- *Speed*: this class contains six possible individuals, namely '*NoSpeed*', '*ExtraSlowSpeed*', '*LowSpeed*', '*NormalSpeed*', '*HighSpeed*' and '*OverSpeed*'. The object's speed is compared with the road's limit to arrive at this value.
- *Acceleration*: this class has five possible values, as follows: '*NoAcceleration*', '*LowDeceleration*', '*LowAcceleration*', '*StrongDeceleration*' and '*StrongAcceleration*'.
- *Weight*: the weight of an object is classified as follows: '*Flyweight*', '*Lightweight*', '*Middleweight*', '*Heavyweight*' and '*SuperHeavyweight*'.

## IV. Case Scenario and Signal Processing

Let us consider a case scenario which will be the basis for the demonstration of the various concepts involved.

### A. Case Scenario

Our sample case scenario is shown in Fig. 6, which shows we have to drive our vehicle in the driving loop. As can be seen, we will encounter rain and fog as we drive, as well as a moving vehicle and static vehicle (a road obstacle) and some pedestrians crossing the street. In two intersections, there are stop signs; lanes for different directions are also shown as well as speed limits to be respected. Here, the safe driving conduct will be tested. A deviation to this will merit the invocation of an assistance mechanism for the driver to drive safely and activation of some signals for the vehicle. A video of this scenario is available online in YouTube website [26].

### B. Driving Simulation

Our driving assistance system concept needs to be tested in the laboratory first before it is tested on the actual roads. To this end, we created our own driving simulator using the software Unity 3D [27]. As shown in Fig. 7, we designed our 3D driving scenario using Unity and the functionalities associated with some moving entities (i.e. vehicle, pedestrians) are implemented using C# programming. As shown, the driving scenario is displayed through three screens positioned next to each other. The driving experience somehow mimics that of the vehicle cockpit. We also bought steering wheels and brake and acceleration pedals for a maximum imitation experience of driving a vehicle. We also incorporate activation of direction signal indicator (i.e. turn right, turn left) and such an indicator is displayed on the screen. Indeed, using this set-up, we are able to drive as if we are driving a real vehicle albeit on a limited scale (the basic driving events: going forward, backward, turn left, turn right, stop, etc.).

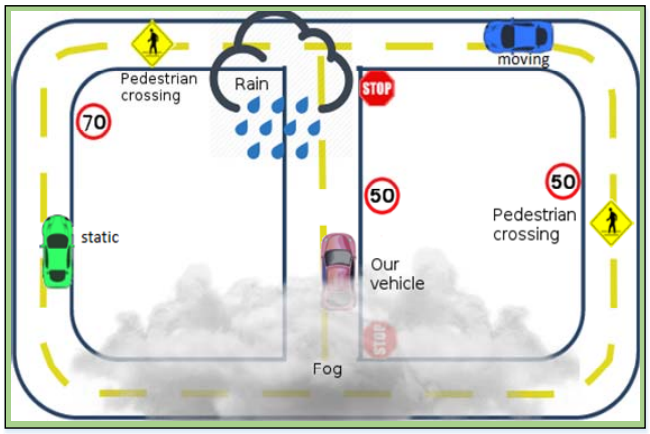



Fig. 6 Sample case scenario

### C. Simulation Data

The identification of a driving event necessitates primarily the collection of various signals coming from the driver, the vehicle and the environment. These signals are then interpreted accordingly. In our work, we collect the following sample representative data from our driving simulator. See Table I.

### D. Simulation Data and the Instantiation of Ontology

The various parameters obtained from the driving simulation are used to instantiate our ontology. We then use a reasoner to check the consistency of the ontology and classify all pour instances. A reasoner is a tool that infers logical consequences from a set of asserted facts. For example, in our simulation snapshot, we create an instance *V* of class Car. In our ontology, we defined class Car as child of class Vehicle. A Car is a vehicle, thus *V* is also a Vehicle. Some reasoner can also apply SWRL rules [28], [29]; this is the case of the Pellet Reasoner [30]. We created a Java program that connects to the Unity 3D driving simulator. We then obtain the csv (comma-separated values) file from the simulation. We match the ontology file with .csv file to instantiate classes, attributes and properties. We then classify and reason using instantiated ontology. The instantiated ontology is saved as a new OWL file [31]. To demonstrate how it all works, consider the turn right in the intersection event with rainy weather (see Fig. 8). The diagram is deceiving but it is actually three screen shots of the event, put side-by-side and the final image is cropped on the right and the left sides. We can see that a signal indicator is activated (i.e. green arrow pointing to the right on the screen). The angle made in turning to the right is *25 degrees* while the speed is quite slow, at *14.56 km/h*.

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

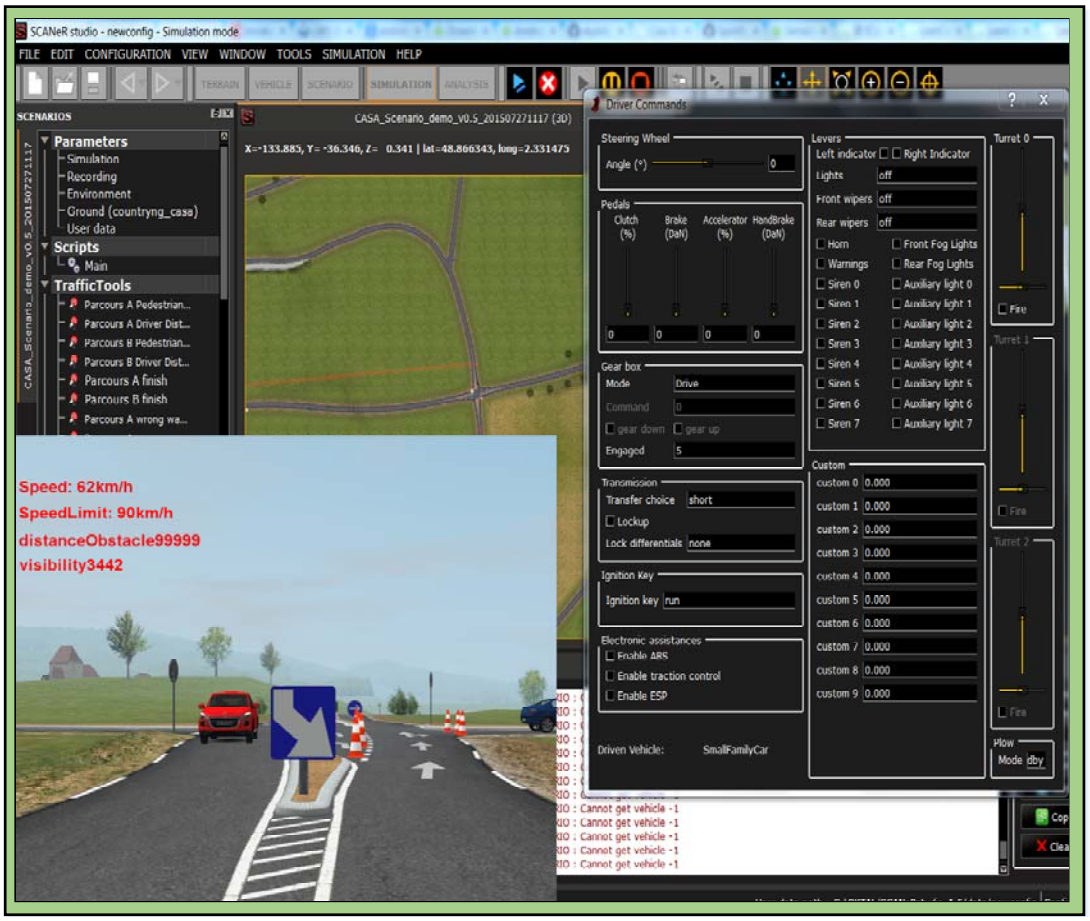



Fig. 7 Our driving simulator

TABLE I
SIMULATION DATA

| Data name | Values | Description |
|---|---|---|
| Image | PNG file | The screenshot of the situation, used to label the data |
| Look orientation | “front”, “left”, “right”, “behind” | Current direction the driver is looking. |
| Steering angle | Float, between 1 and -1 | Current steering wheel angle, -1 is maximum to the left, 1 is maximum to the right, 0 is neutral |
| Throttle | Float between 0 and 1 | Throttle force put on the acceleration pedal, 0 is none, 1 is maximum |
| Brake | Float between 0 and 1 | Brake force put on the braking pedal, 0 is none, 1 is maximum |
| Speed | Km/h | Speed of the vehicle |
| Absolute position | Coordinate on the map formatted as “(X, Y, Z)” with Y being altitude | Position of the vehicle on the map |
| Orientation | “North”, “South”, “East”, “West” | Current orientation of the vehicle |
| Previous Orientation | “North”, “South”, “East”, “West” | Previous orientation of the vehicle |
| Going to | “North”, “South”, “East”, “West” | Orientation needed to travel to the next road segment |
| Size | Positive float value | Size of the vehicle in meters |
| Weight | Positive float value | Weight of the vehicle in kg |
| Speed limit | km/h | Speed limit of the road |
| Blinker (Direction) state | True or False | State of each blinker, true is on and false is off |
| Number of lanes | 0 or positive value | Number of lanes on the current road, 0 if not on the road |
| Continuous line | True , False or nothing | Indicates if the line separating the lanes is continuous or not. No value if we are not on the road |
| Current lane ID | -1 or positive value | ID of the current road segment, -1 if none |
| Next lane | -1 or positive value | ID of the next road segment which is not an intersection, or -1 if none (if we are out of the road for example) |
| Position on lane | -1, 0 or 1 | Current lane we are on. 1 is right lane, -1 is left lane and 0 is not on the road |
| RoadObject | JSON | Road objects such as stop sign, pedestrian and other vehicle that our vehicle is aware of. |
| RoadMap | JSON | Road map that contain all the road, lane, intersection, and the connection between them. |

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

The values obtained from the simulator as the driver performs the turn right event are depicted in Fig. 9. As shown, the class Car is now instantiated with an individual (*My_vehicle*). The various attributes of the Car in relation to *ComponentStatus*, *hasPhysics*, *hasDriver,* is*GoingDirection*, *hasTechnicalData*, *hasDistanceToNextRoadSegment*, etc., properties are also shown. For example, for the hasPhysics property, we can see that the sub-properties have instantiated values as follows: *hasLatitude: 64.4*, *hasLongitude: -46.5*, *speed: low*, *acceleration: low*, *weight of the vehicle: middle weight*, *size of the vehicle: medium*, and the *date of simulation*: *2017-07-04*.

The real context of the driver during simulation is specified as follows: *driver: myself*, *is drunk: false*, *has blood pressure: 10*, *name: Clement*, *age: 23*, *is young driver: false*, etc. The driver's focus on driving is specified as: *focus on driving: focus*, *is looking behind: false*, *is looking to the left: false*, etc.

Likewise, the instantiated ontology for the turn right driving event is shown in Fig. 10. As shown, the individual *My_vehicle* is the instance of class Vehicle. This vehicle is linked to the instances of various classes: Driver (*Myself_driver*), Environment (*My_environment*), Cockpit (*My_cockpit*). It also shows the data property assertions of various attributes of various individuals of various classes. Fig. 11 (a) shows the structure of the roads and their relationship with one another. Fig. 11 (b) shows the individuals in the Protégé tool, depicting the actual relationship of the vehicle and the environment and that of the city and the roads within the environment.

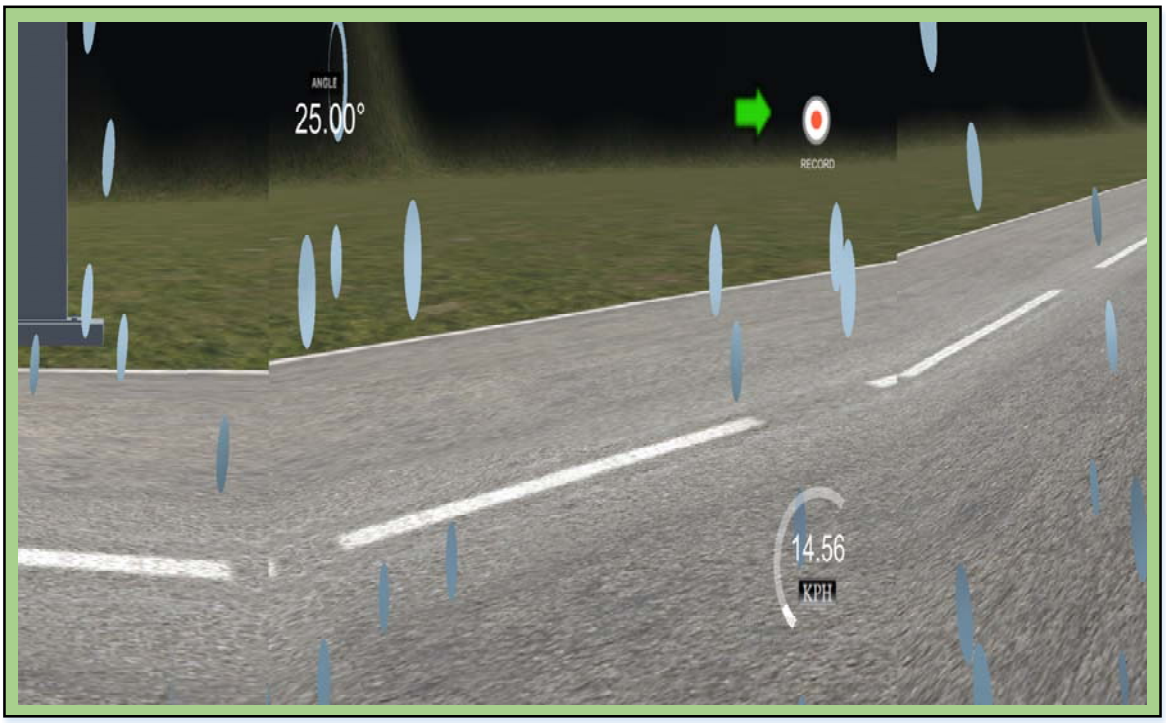



Fig. 8 Turn right driving event on a rainy weather condition

Indeed, we can see smallest details of the vehicle in the ontological representation. Also, it is worth noting that the last two directions of the vehicles are shown in the ontology: *Direction_North* and *Direction_East*. This is in fact a confirmation that prior to turning right (*Direction_East*) in the intersection, the vehicle was heading to the intersection (*Direction_North*).

### *E. Decision Tree Driving Event Classification*

Machine learning algorithms [15], [32]-[36] discover patterns and predict an output from a formatted input after training the algorithm on a sufficiently big set of training data. In our case, the number of possible situations, their variations, and the scale of the data used make this problem appropriate for machine learning. Our problem is a classification problem because we want to be able to give untagged data to our algorithm and get the tag corresponding to the situation.

Decision tree learning [37] uses a decision tree as a predictive model. A decision tree is a flowchart-like structure in which each internal node represents a "*test*" on an attribute, each branch represents the outcome of the test and each leaf represents a class label for the classification tree. A tree can be created by splitting the training set into subset based on an attribute value test and repeating the process until each leaf of the tree contains a single class label or we reach the desired maximum depth. One of advantages of decision tree is that it uses a white-box model. The results given by the model are explained by Boolean logic and are easy to understand, as we can see on the Fig. 12. Here, we can easily see the most important features, i.e. the one that splits the tree in the most meaningful way to separate efficiently all the classes.

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

We used the machine learning algorithms from the *scikit-learn library*. We choose *decision tree* and *k-nearest-neighbor* algorithms because of their speed and simplicity, the possibility of data analysis after learning and because they are the algorithms that gave the best results. We obtained results of *96.18%* of precision for the decision tree algorithm on our test set and *92.65%* with the k-nearest neighbor. Tables II and III show the confusion matrix of each the algorithms. The results indicate a good accuracy although the number of samples is low. For improved results, we need more data with many different variables.

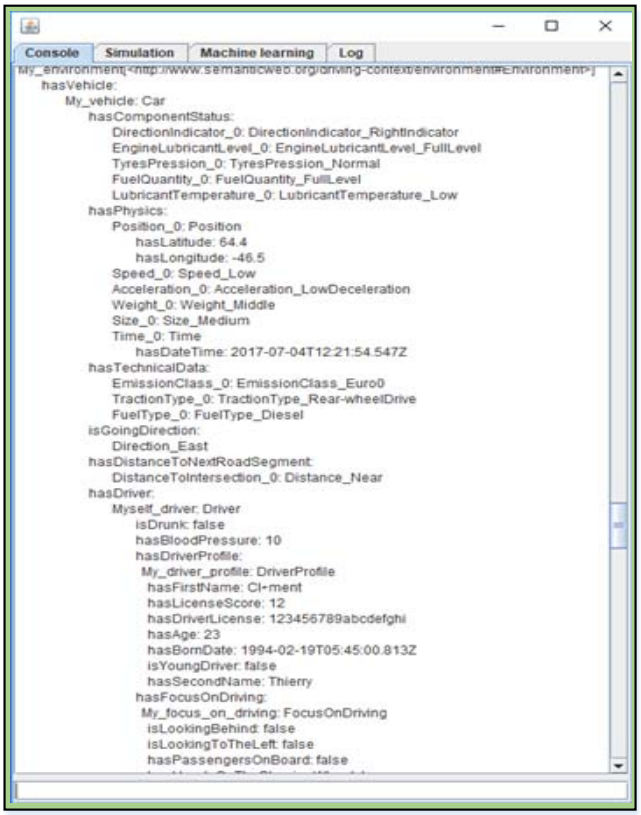



Fig. 9 A driving event produces values that are used to instantiate the ontology template

TABLE II
CONFUSION MATRIX OF DECISION TREE

| | Detected Normal | Detected Stop | Detected Turn left | Detected Turn right |
|---|---|---|---|---|
| Is Normal | 226 | 5 | 1 | 2 |
| Is Stop | 1 | 52 | 0 | 0 |
| Is Turn left | 2 | 0 | 25 | 1 |
| Is Turn right | 1 | 0 | 0 | 24 |

TABLE III
CONFUSION MATRIX OF NEAREST NEIGHBOR

| | Detected Normal | Detected Stop | Detected Turn left | Detected Turn right |
|---|---|---|---|---|
| Is Normal | 228 | 1 | 3 | 2 |
| Is Stop | 1 | 51 | 1 | 0 |
| Is Turn left | 6 | 1 | 16 | 5 |
| Is Turn right | 4 | 1 | 0 | 20 |

### *A. Validating Connection between Simulation and Machine Learning*

To connect the simulation to the model, we use the TCP protocol. A python server is created and waits for a connection from the simulator. Once the simulation is connected, the server waits for an input from the simulation. The simulation sends data of the current driving event, .csv formatted. The server then processes the data received, and then use them with the model it saved to predict which driving event corresponds to the sampled data. After predicting the result, the server displays messages (see Fig. 13) and sends back the current situation to the simulator in order to implement the necessary action that corresponds to the situation. Related message/s intended for driver is then displayed on the screen.

Fig. 14 shows two representative sample messages intended for the driver based on the given driving event. Fig. 14 (a) shows an over speeding message (i.e. the driver's speed is 68.07 km/h while the road's speed limit is 50 km/h). Fig. 14 (b) shows a message informing the driver to stay on the lane.

## V. CONCLUSION

In this paper, we have modeled the driving context using ontological approach. Our goal is to determine driving context as it happens, classify it and provide assistance in situations where driver needs to be informed or alerted. This phase of our work is the preliminary phase where simple and common driving events are taken into account. We will proceed next to more complex driving events in the next phase of the project. Our ultimate aim is to assist drivers towards safe driving, green driving and comfortable driving. The level of assistance is of three types: notification, alert and action towards the vehicle. This is an ongoing work and it will evolve further in the days ahead. Future works include the reinforcement learning to incorporate safe, green and comfortable driving features for every driving event, the cognitive user interface design [38], [39] and the cognitive component [40], [41] that allows our system to learn new driving situation, reason with purpose and interact with humans naturally. It will learn from its interaction with the system users and from its experiences with its environment.

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

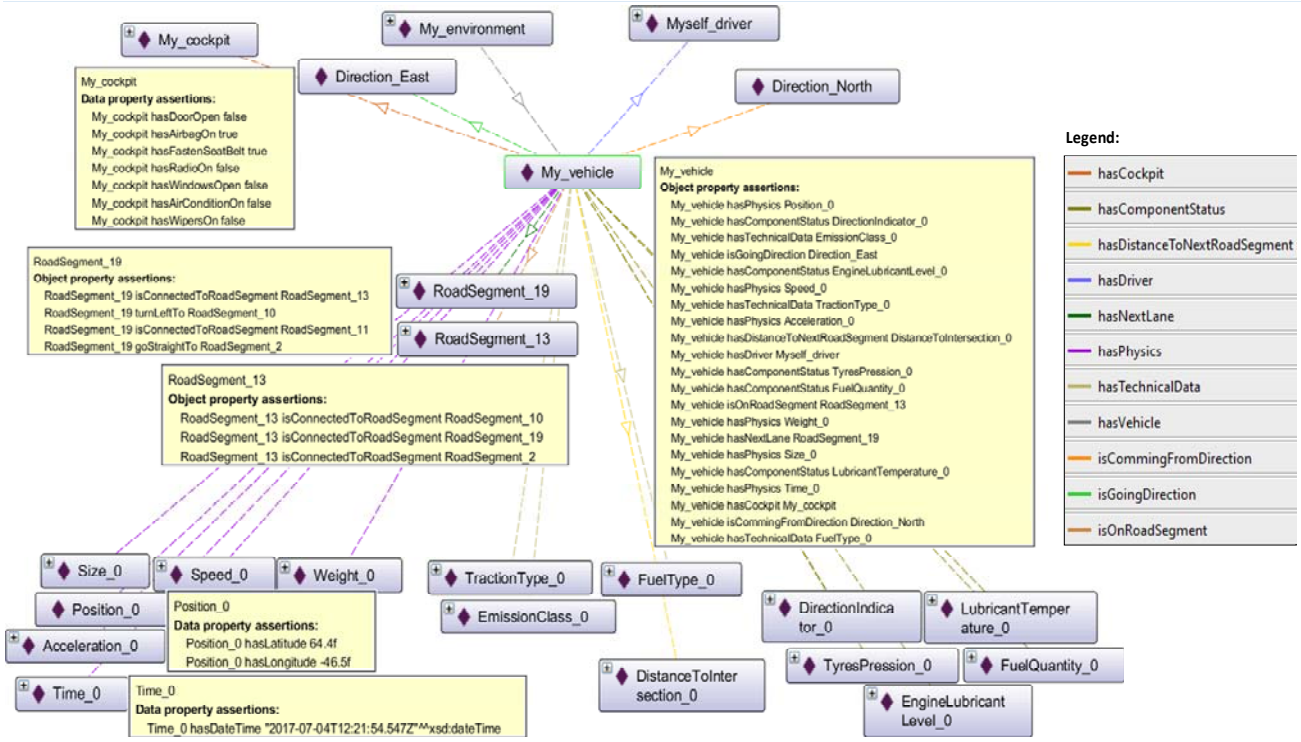



Fig. 10 The instantiated ontological representation of the context of the vehicle for the turn right driving event

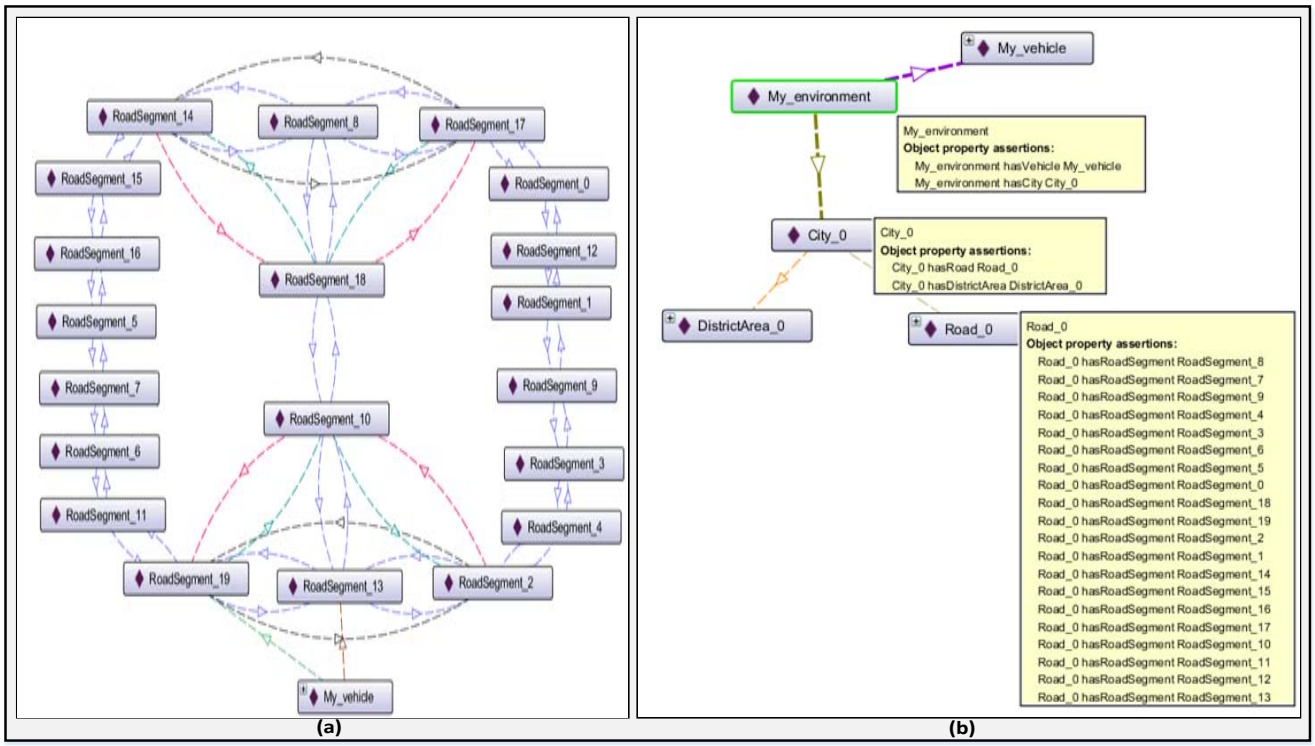



Fig. 11 (a) The ontological representation of road segments. (b) The individuals of various classes (vehicle, and individuals for various environment classes) for the simulated turn right driving event

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

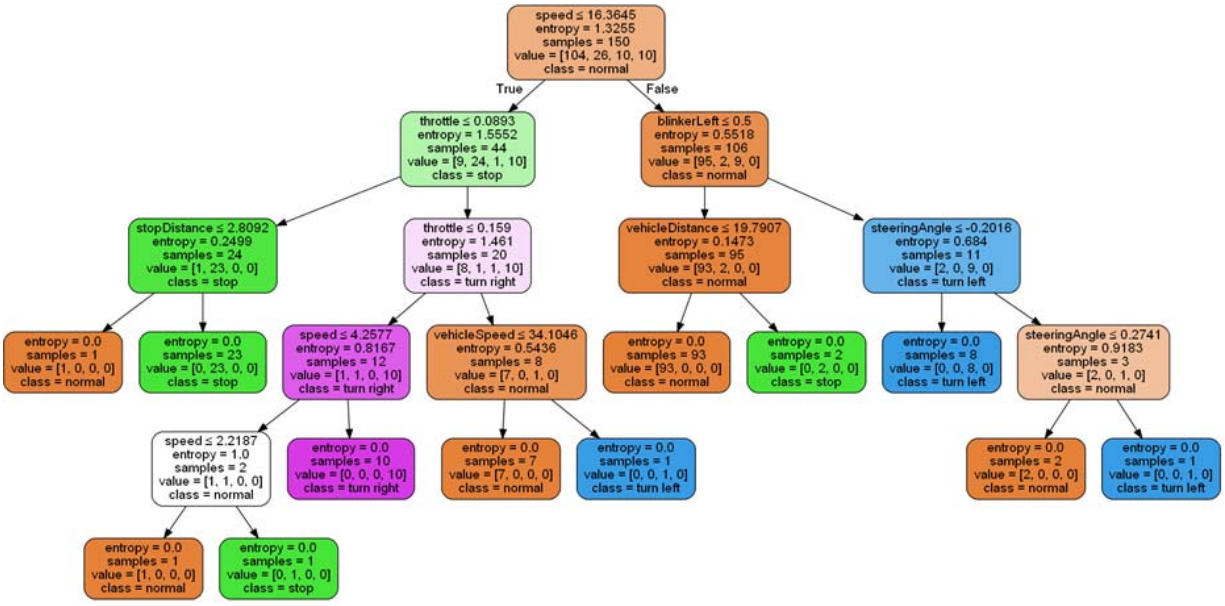



Fig. 12 Classification tree example

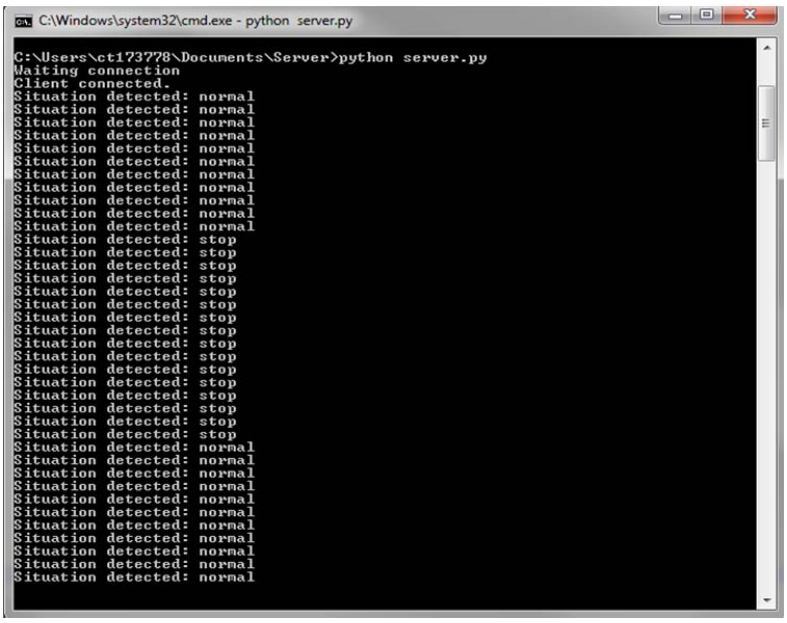



Fig. 13 The server detecting situations

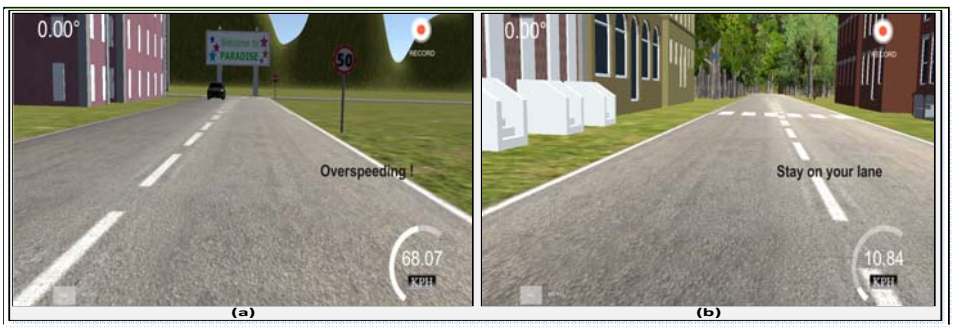



Fig. 14 (a) Driving assistance message for over speeding (b) Driving assistance message reminding driver to stay on the lane

Open Science Index, Computer and Information Engineering Vol:12, No:2, 2018 publications.waset.org/10008483/pdf

## REFERENCES

[1] Dice. (2017). Ontologies. Available: https://www.dice.com/skills/Ontologies.html
[2] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, et al. (1991) Enabling Technology for Knowledge Sharing. AI Magazine. 36-56.
[3] N. Guarino, "Formal ontology, conceptual analysis and knowledge representation," Human-Computer Studies, vol. 43, pp. 625-640, 1995.
[4] T. R. Gruber, "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition, vol. 5, pp. 199 - 220, 1993.
[5] A. Maalel, H. H. Mabrouk, L. Mejri, and H. H. B. Ghezela, "Development of an ontology to assist the modeling of accident scenario application on railroad transport," Journal of Computing, vol. 3, 2011.
[6] M. Charest and S. Delisle, "Ontology-Guided Intelligent Data Mining Assistance: Combining Declarative and Procedural Knowledge," presented at the IASTED International Conference, 2006.
[7] S. Kannan, A. Thangavelu, and R. Kalivaradhan, "An Intelligent Driver Assistance System (I-DAS) for Vehicle Safety Modeling Using Ontology Approach," International Journal of Ubiquitous Computing, vol. 1, pp. 15 – 29, 2010.
[8] Protégé. (2016). Protégé: Open-source ontology editor. Available: http://protege.stanford.edu
[9] VOWL. Visual Notation for OWL Ontologies. Available: http://vowl.visualdataweb.org/v2/
[10] H. Abraham, C. Lee, S. Brady, C. Fitzgerald, B. Mehler, B. Reimer, et al., "Autonomous Vehicles, Trust, and Driving Alternatives: A survey of consumer preferences," Massachussets Institute of Technology AgeLab, Cambridge, MA, USA, 2016.
[11] Foley & Lardner LLP, "2017 Connected Cars & Autonomous Vehicles Survey," 2017. Available: https://www.foley.com/files/uploads/2017-Connected-Cars-Survey-Report.pdf
[12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements and future directions," Elsevier Future Generation Computer Systems, vol. 29, pp. 1645-1660, 2013.
[13] G. Brioschi, M. D. Hina, A. Soukane, A. Ramdane-Cherif, and M. Colombetti, "Computational intelligence on the cognition of driving context for a safe driving assistance system," International Journal of Software Science and Computational Intelligence, vol. 9, 2017.
[14] G. Brioschi, M. D. Hina, A. Soukane, A. Ramdane-Cherif, and M. Colombetti, "Techniques for Cognition of Driving Context for Safe Driving Application," presented at the ICCI*CC 2016, 15th IEEE International Conference on Cognitive Informatics and Cognitive Computing Stanford, CA, USA, 2016.
[15] P. Tchankue, J. Weeson, and D. Vogts, "Using machine learning to predict the driving context whist driving," in SAICSIT '13 South African Institute for Computer Scientists and Information Technologists Conference, East London, South Africa 2013.
[16] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," presented at the ITSC 2015, IEEE International Conference on Intelligent Transportation Systems, Canary Islands, Spain, 2015.
[17] A. Armand, D. Filliat, and J. Ibañez-Guzman, "Ontology-Based Context Awareness for Driving Assistance Systems," presented at the IEEE Intelligent Vehicles Symposium, Dearborn, MI, USA, 2014.
[18] S. Fernandez, R. Hadfi, T. Ito, I. Marsa-Maestre, and J. R. Velasco, "Ontology-Based Architecture for Intelligent Transportation Systems Using a Traffic Sensor Network," Sensors, vol. 16, 2016.
[19] P. Morignot and F. Nashashibi, "An ontology-based approach to relax traffic regulation for autonomous vehicle assistance," presented at the AIA'13, 12th IASTED International Conference on Artificial Intelligence and Applications, Innsbruck, Austria.
[20] M. Boudra, M. D. Hina, A. Ramdane-Cherif, and C. Tadj, "Architecture and Ontological Modelling for Assisted Driving and Interaction," International Journal of Advanced Computer Research, vol. 5, 2015.
[21] Fraunhofer IAO. (2016). Human-Vehicle Interaction: From driver assistance system to automated driving. Available: http://www.iao.fraunhofer.de/lang-en/range-of-services/people-and-mobility/human-vehicle-interaction.html
[22] Z. Xiong, V. Dixit, and T. Waller, "The Developement of an Ontology for Driving Context Modelling and Reasoning," presented at the 19th IEEE International Conference on Intelligent Transportation Systems, Rio de Janiero, Brazil, 2016.
[23] S. Dourlens and A. Ramdane-Cherif, "Modeling and understanding environment using semantic agents," International Journal of Artificial Intelligence and Soft Computing, vol. 2, 2012.
[24] P. P. Daghan Lemi Acay, Liz Sonenberg, "Extrospection: Agents Reasoning about the Environment," presented at the 3rd International Conference on Intelligent Environments (IE'07), Germany, 2007.
[25] S. Dourlens and A. Ramdane-Cherif, "Semantic Modeling and Understanding of Environment Behaviors," presented at the IEEE Symposium Series on Computational Intelligence, Symposium on Intelligent Agents, Paris, France, 2011.
[26] C. Thierry. (2017). Project ASSIA Simulation. Available: https://www.youtube.com/watch?v=pH71u6a6HYc&feature=youtu.be
[27] Game Engine. (2016). Unity 3D. Available: https://unity3d.com/
[28] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. (2016). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Available: https://www.w3.org/Submission/SWRL/
[29] S. Julien and P. Maret, "Semantic Agent Model for SWRL Rule-based Agents," in International Conference on Agents and Artificial Intelligence (ICAART 2010), Valencia, Spain, pp. 244-248.
[30] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A Practical OWL-DL Reasoner," Journal of Web Semantics, vol. 5, pp. 51 - 53, 2007.
[31] B. Motik, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, OWL 2 Web Ontology Language Profiles, 2nd edition, 2012.
[32] I. Witten, E. Frank, and M. Hall, Data Mining: Practical Machine Learning Tools and Techniques. Burlington, MA: Morgan Kaufmann, 2011.
[33] A. A. Ithape, "Artificial Intelligence and Machine Learning in ADAS," presented at the Vector India Conference 2017, Pune, India, 2017.
[34] P. Louridas and C. Ebert, "Machine Learning," IEEE Software, vol. 33, pp. 110-115, Sept.-Oct. 2016 2016.
[35] RapidMiner Inc., "How to Correctly Validate Machine Learning Models," 2017.
[36] A. Saxena. (2017, 27 September 2017). Machine Learning Algorithms in Autonomous Cars. Available: http://visteon.bg/2017/03/02/machine-learning-algorithms-autonomous-cars/
[37] Y. Hou, P. Edara, and C. Sun, "Modeling Mandatory Lane Changing Using Bayes Classifier and Decision Trees," IEEE Transactions on Intelligent Transportation Systems, vol. 15, pp. 647 - 655, 2014.
[38] M. F. Peschl and C. Stary, "The Role of Cognitive Modeling for User Interface Design Representations," Mind and Machines, vol. 8, pp. 203 - 236, 1998.
[39] T. Harada, K. Mori, A. Yoshizawa, and H. Iwasaki, "Designing a Car-Driver's Cognitive Process Model for Considering Degree of Distraction," International Journal of Software Science and Computational Intelligence, vol. 7, pp. 1-16, July-September 2015 2015.
[40] John. E. Kelly III, "Computing, cognition and the future of knowing: How humans and machines are forging a new age of understanding," 2015.
[41] L. Li, D. Wen, N.-N. Zheng, and L.-C. Shen, "Cognitive Cars: A New Frontier for ADAS Research," IEEE Transactions on Intelligent Transportation Systems, vol. 13, pp. 395 - 407, 2012.