Explaining Traffic Situations – Architecture of a Virtual Driving Instructor

Martin Sandberg, Johannes Rehm, Matěj Mňouček, Irina Reshodko and Odd Erik Gundersen

April 20, 2020

Explaining Traffic Situations – Architecture of a Virtual Driving Instructor*

Martin K. H. Sandberg[1], Johannes Rehm[1,2](✉), Matej Mnoucek[1], Irina Reshodko[2], and Odd Erik Gundersen[1]

[1] Department of Computer Science, Norwegian University of Science and Technology, Trondheim, Norway
{johannes.rehm, odderik}@ntnu.no
martin.hoel.sandberg@gmail.com
[2] Way AS, Trondheim, Norway
irina@way.no

Abstract. Intelligent tutoring systems become more and more common in assisting human learners. Distinct advantages of intelligent tutoring systems are personalized teaching tailored to each student, on-demand availability not depending on working hour regulations and standardized evaluation not subjective to the experience and biases of human individuals. A virtual driving instructor that supports driver training in a virtual world could conduct on-demand personalized teaching and standardized evaluation. We propose an architectural design of a virtual driving instructor system that can comprehend and explain complex traffic situations. The architecture is based on a multi-agent system capable of reasoning about traffic situations and explaining them at an arbitrary level of detail in real-time. The agents process real-time data to produce instances of concepts and relations in an ever-evolving knowledge graph. The concepts and relations are defined in a traffic situation ontology. Finally, we demonstrate the process of reasoning and generating explanations on an overtake scenario.

Keywords: Virtual driving instructor · Intelligent tutoring system · Situation awareness · Multi-agent system · Ontology · First order logic · Explanations

1 Introduction and Related Work

The field of Intelligent Tutoring System (ITS) has matured since the conception of the idea in 1970s [20], and ITS implementations have been used in various domains of life, such as crisis management [15] and vehicle driving training [2]. The main goal of ITS is to support effective learning and reduce workload of human teachers. In order to do that, an ITS must contain all the concepts, rules and decision-making approaches necessary for the domain awareness – in other

* This research was funded by Way AS and the Norwegian Research Council through the TRANSPORT program under the grant agreement 296640.

words, it has to have an adequate *domain model*. It is also crucial to have a model of the cognitive state of the student and the learning tendencies – that is a *student model*. Finally, an ITS has to employ a *tutoring model* in order to choose the right form and timing of teaching feedback [20]. A very important additional requirement for an ITS is its explainability – it should be clear why the system made a decision. The sophistication of the domain comprehension necessary for effective teaching in most cases requires the use of an artificial intelligence (AI) system. According to Gunning and Aha, explainable AI is "AI systems that can explain their rationale to a human user, characterise their strengths and weaknesses, and convey an understanding of how they will behave in the future" [9]. Particularly, following correct traffic rules and the ability to produce explanations requires an adequate awareness about the situation [13]. Situation awareness (SAW) is obtained by perceiving all relevant elements in the environment, understanding their meaning and predicting their future state [5]. An ITS should be able to explain to the student how he or she handled a certain traffic situation, therefore it needs situation awareness and explainability.

One of the most widely used methods in AI, deep neural networks [8], have achieved tremendous results during the last decade. However, neural networks are inherently black box systems, and do not maintain an explicit awareness of a situation. Although there are attempts to make certain aspects of these systems explainable, such as deep explanations [7] and model induction [11], complete and interpretable explanations are still problematic. This is mainly due to the vast amount of operations performed in a deep neural network. The explainability requirement for ITS thus calls for the use of interpretable-by-design AI approaches. Ontologies that describe explicit relations between relevant concepts have been widely used to represent situations conceptually [15]. The multi-agent paradigm together with rule-based reasoning on an ontology has proven to be highly suitable for representing and reasoning with traffic situations [4].

Our work presents an architectural design of a virtual driving instructor (VDI) system, which uses the data from a virtual reality (VR) driving simulator in order to provide comprehensive integrated teaching assistance both online (during the driving lesson) and offline (as an after-lesson debrief). The virtual driving instructor should be able to reason about complex situations, i.e. situations that comprise of several sub-situations. Driving on public roads requires to perform multiple tasks in parallel, such as staying in the lane and maintaining a safe distance to the cars in front. Thus, the VDI system must be able to continuously assess several sub-situations at the same time. The explanation data which this system produces has to be as complete as possible for the traffic domain, and thus allow for a feedback of any level of detail.

Buechel *et al.* [3] propose an ontological framework for reasoning about various traffic scenarios which can be easily generalised to different traffic rule contexts. Their work focuses on regulation-aware decision-making for autonomous cars but lacks details and temporal aspects necessary for explanation purposes in our proposed ITS. A traffic domain ontology was presented by Zhao *et al.*

[22], and was tested to solve the yielding problem in narrow and uncontrolled intersections.

Zamora *et al.* [21] introduce a rule-based multi-agent architecture for an intelligent ADAS system assisting a driver in an urban environment. The main goal of the work by Zamora *et al.* is to recognise and display a warning about a potentially dangerous situation which the driver is not aware of, so it has no need for complex back in time reasoning or detailed explanations. This proposal has later been implemented and experimentally validated in [17]. Both works are based on the multi-agent approach described in [10]. The blackboard-based CarCOACH system developed by Arroyo *et al.* [1] uses a multi-agent architecture to assess scenarios such as hard braking. The CarCOACH system focuses on detecting and gently coaching the driver through immediately dangerous behaviour such as speeding while turning, and does not address more complicated scenarios or rule compliance. Sukthankar *et al.* [18] proposed and validated a multi-agent system with arbitrated voting for automated vehicle decision-making on a highway. As a vote-based system, it is not designed to provide a detailed explanation of its decisions. Weevers *et al.* [19] present a high-level multi-agent architecture of a virtual driving instructor for a commercial driving simulator. It can recognise and evaluate speed control and intersection handling, and is capable of reasoning if the student has performed certain driving tasks correctly with respect to the situation. However, the details of the architecture are not disclosed.

It is worth noting that an ontology-based multi-agent architecture is not the only way to approach the SAW problem. Raptis *et al.* [16] develop an attentiveness assessment system which tracks the hand gestures of a driver on the steering wheel using an SVM-based method to estimate an attention score. Many more approaches [14] exist for recognising and evaluating driving manoeuvres and behaviour.

The rest of the paper is structured as follows: In section 2, we present our proposal of the traffic situation modelling, and in subsection 2.3 we talk about generation of explanations. The framework is illustrated by a detailed use case in section 3. Finally, we conclude in section 4.

2 Modelling Traffic Situations

2.1 Ontology

We use an ontology, illustrated in Figure 1, to design the situation model. The ontology is based on Matheus, Kokar and Baclawski [12]. This work provides a more thorough explanation. The central part of this ontology is the *SituationObject* which can be either a *PhysicalObject*, e.g. *Car*, or a *NonPhysicalObject*, e.g. *Lane*. *Situation* extends *SituationObject* which allows to model a situation as an aggregation of sub-situations. *SituationObject* can have multiple instances of *Attribute* like speed or brake pressure of a car. In contrast to *Attribute*, which is specific to a *SituationObject*, *Relation* defines a relation between two situation

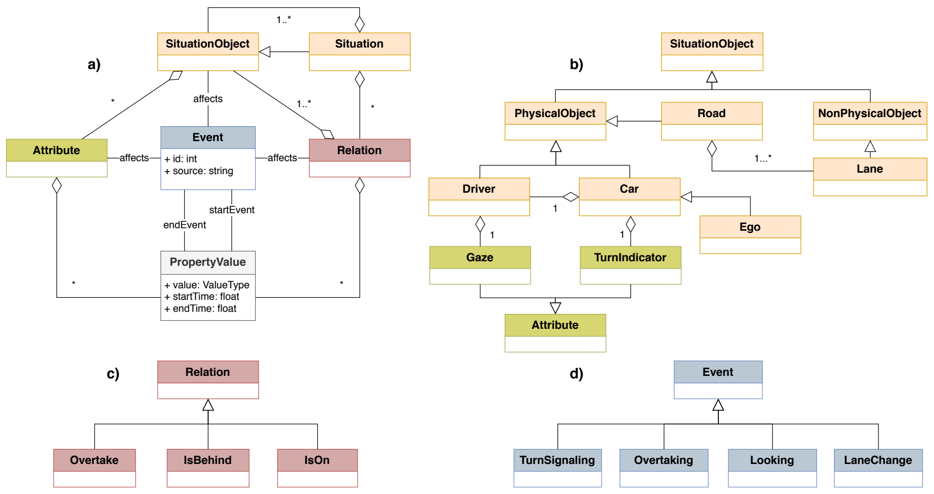

Fig. 1. (a) The core ontology introduced by Matheus *et al.* [12] is the basis of the domain-specific ontology proposed in this work. The expansion of **(b)** *SituationObject* and *Attribute*, **(c)** *Relation* and **(d)** *Event*. The concepts are expanded to sufficiently cover a simplified overtake scenario, see section 3.

objects. For example *CarA* is located on (*isOn*) *LaneX*. However, the values of attributes and relations can change over time. This is captured by the *PropertyValue* concept.

Individual attributes and relations do not have a single value assigned to them, but multiple instances of *PropertyValue* with non-overlapping time intervals. The *Event* concept indicates a change of an attribute, relation, or more abstractly change in the situation, e.g. a change of *isOn* relation from *LaneX* to *LaneY* would generate a *LaneChange* event. A new *PropertyValue* is created whenever a new *Event* occurs.

In cases where we want to reason back in time, the actual event happens at or after the end time of another property value. We extend the core ontology of [12] in three aspects to be able to reason back in time. 1) The start and end time of a *PropertyValue* is not automatically set by the time of the current and next event. They can be inferred and set individually, contrasting to the core ontology of [12] where two temporally adjacent instances of *Event* always set the *StartTime* and the *EndTime* of a *PropertyValue*. 2) Start and end time need to be non-overlapping, but there can be a gap in between the end time of one property value and the start time of the next property value. 3) Situation objects, attributes and relations can be added dynamically once they get relevant for the situation.

As traffic situations can get very complex, the provided version of ontology is simplified and consists only of the objects relevant for the presented overtake scenario, shown in section 3.

2.2 Multi-Agent System

We employ a multi-agent system to assess all situations which arise during a lesson. Each agent performs specified tasks and communicates with the others through a blackboard, which stores all the data generated by agents. We differentiate between two types of agents: property value and explanation. A property value agent produces property values in the form of attributes or relations, and generates events accordingly when these attributes and relations change. Additionally, a property value agent can dispatch execution triggers to other agents. An explanation agent assesses if the student behaves correctly in a situation, and generates situation-specific explanations. In Figure 2 we show a schematic of our suggested multi-agent system. It is constrained to assess a simplified overtake scenario for clarity. The agents are arranged in a hierarchical structure. Low-level agents perform basic assessment, such as checking the gaze of the driver or which lane the car is on. High-level agents can perform comprehensive tasks, for example assessing a complete overtake manoeuvre using data generated by many lower-level agents. Most explanation agents are thus higher-level agents.

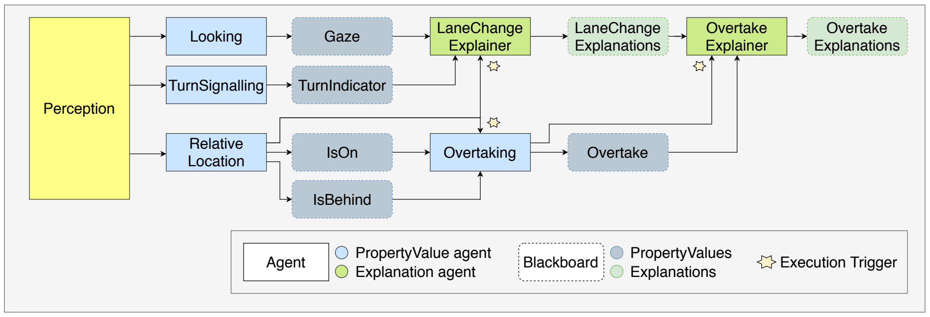

Fig. 2. The schematic of the proposed multi-agent system, simplified for a reduced overtake scenario. The *RelativeLocationAgent* dispatches execution triggers whenever a lane change has occurred.

2.3 Generating Explanations

Explanations play an important part in teaching. High quality explanations creates trust and motivate students. Our goal is to design a framework which generates explanations about both right and wrong behaviour at traffic situations encountered by students. This requires a system that has the domain knowledge to be able to assess the situation and the decisions the student made.

In our architecture, the domain knowledge is encoded within the ontology and the agents. Explanation agents are triggered by property value agents in order to infer if traffic rules were followed and if the driver behaved properly in the given situation. The output of an explanation agent is a recursive vector structure containing all information necessary, also from more low-level explanation agents, for feedback generation. Each entry of the output vector contains an appropriate value or another vector which is the output of a subordinate agent. This structure allows to give explanations at any level of detail and is by itself already explainable. But the task of generating feedback for the student requires appropriate conversion of this data structure to a human-understandable form. The type of feedback which is suitable in each particular situation depends on whether it should be given online (while the lesson is running) or offline (as a debriefing session after the lesson). The online feedback is typically given as a short audio statement or an icon overlay, or via objects in the simulation itself. The offline feedback which is not time-sensitive can be more detailed. A detailed explanation text or an annotated video recording are examples of offline feedback. In this work, we focus on natural language feedback provided in an offline setting. The explanation vectors we propose have clearly defined structure and relatively small range of possible values. This allows us to use a simple and more robust template approach [6]. An example on how to generate text from explanation vectors is given in subsection 3.3.

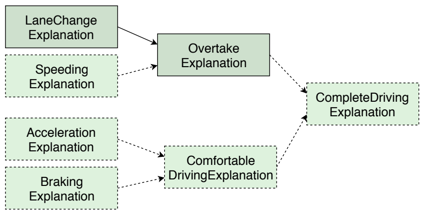

Fig. 3. Explanation tree structure illustrating that all explanations culminate into a high level *CompleteDriving-Explanation*. The figure shows the possibility of expanding the tree when new explanations are needed. It is not restricted to a binary tree. In this paper, we use only the *LaneChangeExplanation* and the *OvertakeExplanation*.

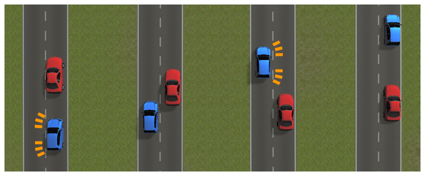

Fig. 4. Visualisation of the progress of the scenario, from left to right: indicate the intention to overtake $\longrightarrow$ change lanes and pass the car $\longrightarrow$ indicate the intention to return to the original lane $\longrightarrow$ change back to the original lane.

3 Scenario: Overtake

3.1 Scenario Development

In the following, we outline a scenario in which the Ego car, driven by the driving student, overtakes another car, as depicted in Figure 4. For this simplified case, we concentrate on the correct usage of turn signals and side mirror observation.

While the scenario develops over time, the situation is observed by adding property values to the ontology describing the complete situation. As stated before, new property values record relations between situation objects and attributes of situation objects within a time period. Property values for relations with a boolean value and attributes are denoted as

$$< \mathit{relation} > (\mathit{SituationObject}_x, \mathit{SituationObject}_y, < \mathit{start\ time} >)$$
$$< \mathit{attribute} > (\mathit{SituationObject}_x, < \mathit{start\ time} >) :=< \mathit{value} >$$

Scenario:

- *Event* t_0: Start situation
  - $\mathit{IsOn}(\mathbf{Ego}, \mathbf{RightLane}, t_0)$
  - $\mathit{IsOn}(\mathbf{Car}, \mathbf{RightLane}, t_0)$
  - $\mathit{IsBehind}(\mathbf{Ego}, \mathbf{Car}, t_0)$
- *Event* t_1: Signalling left turn
  - $\mathit{TurnIndicator}(\mathbf{Ego}, t_1) := \text{Left}$
- *Event* t_2: Looking at left mirror
  - $\mathit{Gaze}(\mathbf{Driver}, t_2) := \text{LeftMirror}$
- *Event* t_3: Changing lanes
  - $\mathit{IsOn}(\mathbf{Ego}, \mathbf{LeftLane}, t_3)$
- *Event* t_4: In front of other Car
  - $\mathit{IsBehind}(\mathbf{Car}, \mathbf{Ego}, t_4)$
- *Event* t_5: Signalling right turn
  - $\mathit{TurnIndicator}(\mathbf{Ego}, t_5) := \text{Right}$
- *Event* t_6: Looking at right mirror
  - $\mathit{Gaze}(\mathbf{Driver}, t_6) := \text{RightMirror}$
- *Event* t_7: Changing lanes
  - $\mathit{IsOn}(\mathbf{Ego}, \mathbf{RightLane}, t_7)$
  - $\mathit{Overtake}(\mathbf{Ego}, \mathbf{Car}, t_7)$
- *Event* t_8: End situation
  - $\mathit{TurnIndicator}(\mathbf{Ego}, t_8) := \text{Off}$

3.2 Reasoning Using First Order Logic

To identify state changes in attributes and relations, agents uses first-order logic as the predicates shown below.

$$\begin{aligned}&\neg \mathit{IsOn}(\mathbf{Ego}, \mathbf{Lane}, t-1) \wedge\\ &\mathit{IsOn}(\mathbf{Ego}, \mathbf{Lane}, t)\\ &\implies \mathit{LaneChange}(\mathbf{Ego}, \mathbf{Lane}, t)\end{aligned}$$

$$\begin{aligned}&\neg\,(\mathit{TurnIndicator}(\mathbf{Ego}, t-1) = \text{Left}) \wedge\\ &(\mathit{TurnIndicator}(\mathbf{Ego}, t) = \text{Left})\\ &\implies \mathit{TurnSignalling}(\mathbf{Ego}, \text{Left}, t)\end{aligned}$$

The *OvertakeAgent* has one task – to recognise an overtake event. An overtake should be checked every time Ego does a lane change. The *LaneChange* event triggers the execution of the *OvertakeAgent*. Additionally, we query about the temporal data.

$$LOT(\mathit{car}, \mathit{lane}, t_{cur}) \equiv \operatorname*{argmin}_{\forall t : \exists \mathit{IsOn}(\mathit{car}, \mathit{lane}, t) \wedge t < t_{cur}} (t_{cur} - t) \tag{1}$$

To know if an overtake occurred, one would like to know when the last occurrence in time (*lot*) where Ego was in the same lane as the one it is currently on. We can find *lot* using Equation 1, $lot = LOT(\mathbf{Ego}, \mathbf{Lane}, t)$, and input it into the overtake rule as follows.

$$\begin{aligned}
&\mathit{IsOn}(\mathbf{Ego}, \mathbf{Lane}, t) \wedge \mathit{IsOn}(\mathbf{Car}, \mathbf{Lane}, t) \wedge \mathit{IsBehind}(\mathbf{Car}, \mathbf{Ego}, t) \wedge \\
&\mathit{IsOn}(\mathbf{Ego}, \mathbf{Lane}, lot) \wedge \mathit{IsOn}(\mathbf{Car}, \mathbf{Lane}, lot) \wedge \mathit{IsBehind}(\mathbf{Ego}, \mathbf{Car}, lot) \\
&\implies \mathit{Overtake}(\mathbf{Ego}, \mathbf{Car}, t)
\end{aligned}$$

3.3 Explanations

By explicitly utilising explanation agents, one is able to separate the situation comprehension and the situation explanation. For instance, the *RelativeLocationAgent* recognises that a lane change has occurred, but does not care about how well the lane change was done. That task is delegated to the *LaneChangeExplainerAgent*.

The *RelativeLocationAgent* notifies the *LaneChangeExplainerAgent*, and the explanation agent assesses if the driver remembered to conduct all of the important actions needed for a proper lane change. To accomplish this, it need access to attributes from the past such as the property values *TurnIndicator* and *Gaze*. To confirm that the driver looked at the correct mirrors and switched on the turn signals prior to the time of the lane change, $t_{LaneChange}$, we define a time interval of interest as $I \equiv [t_{LaneChange} - i, t_{LaneChange}]$. Here i is a natural number defining the duration of the time interval in abstract units. Additionally, one could check if the sequence of these behaviours were correct, but in this example the explanation will be kept on a basic level. The *LaneChangeExplainerAgent* checks the following rules whenever a lane change occurs.

$$\begin{aligned}
&\text{SignalLeft} := \exists \mathit{TurnIndicator}(\mathbf{Ego}, t) = \text{Left} : t \in I \\
&\text{LeftMirrorLook} := \exists \mathit{Gaze}(\mathbf{Driver}, t) = \text{LeftMirror} : t \in I \\
&\text{LaneChangeExplanation} := \{\text{SignalLeft}, \text{LeftMirrorLook}\}
\end{aligned}$$

The results of these predicates are stored in the *LaneChangeExplanation* vector. Hence, given a lane change explanation, $LCE := \{\text{True}, \text{False}\}$, a template approach can be applied in the natural language generation.

You performed a lane change.
You did turn signal < *$TurnIndicator* ? correctly : incorrectly >,
and you did < *$SideMirrorLook* ? check : not check >
for cars behind you in the side mirror.

The explanation tree, illustrated in Figure 3, shows that our multi-agent system can provide a complete explanation of a high level situation using the recursive vector structure. Multiple text snippets, from each explanation, can be merged to form a detailed human readable explanation. In this case, the *OvertakeExplainerAgent* has access to the *LaneChangeExplanations*. By retrieving the two lane change explanations (LCE_1 and LCE_2) which defined the overtake, one can explain the complete overtake process.

$$\text{OvertakeExplanation} \equiv \{LCE_1, LCE_2\}$$

4 Conclusion

We have developed an architecture for a virtual driving instructor system, which can assess complex situations, such as the presented overtake scenario, and can derive conclusions or explanations of interest.

The multi-agent system allows the VDI to recognise traffic regulation violations as well as correct traffic behaviour. The explanation data structure generated by the multi-agent system has all the information necessary to generate complete, interpretable and traceable explanations. An example of such explanation using templates is also shown for the example of an overtake scenario.

As this work focuses on architectural design of an ITS, its implementation is a necessary next step in this project. An integrated ITS also requires a detailed design of the student model and its relation with the personalized feedback concept. Development of the student model is also left as a future work.

References

1. Arroyo, E., Sullivan, S., Selker, T.: CarCOACH: A polite and effective driving COACH. Conference on Human Factors in Computing Systems - Proceedings pp. 357–362 (2006). https://doi.org/10.1145/1125451.1125529
2. Backlund, P., Engström, H., Johannesson, M., Lebram, M.: Games for traffic education: An experimental study of a game-based driving simulator. Simulation and Gaming **41**(2), 145–169 (2010). https://doi.org/10.1177/1046878107311455
3. Buechel, M., Hinz, G., Ruehl, F., Schroth, H., Gyoeri, C., Knoll, A.: Ontology-based traffic scene modeling, traffic regulations dependent situational awareness and decision-making for automated vehicles. In: 2017 IEEE Intelligent Vehicles Symposium (IV). vol. 7, pp. 1471–1476. IEEE (jun 2017). https://doi.org/10.1109/IVS.2017.7995917
4. Chen, B., Cheng, H.H.: A review of the applications of agent technology in traffic and transportation systems. IEEE Transactions on Intelligent Transportation Systems **11**(2), 485–497 (2010). https://doi.org/10.1109/TITS.2010.2048313
5. Endsley, M.R.: Toward a theory of situation awareness in dynamic systems. Human Factors **37**(1), 32–64 (1995). https://doi.org/10.1518/001872095779049543
6. Gatt, A., Krahmer, E.: Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. CoRR **abs/1703.09902** (2017)
7. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining explanations: An overview of interpretability of machine learning. In: Proceedings - 2018 IEEE 5th International Conference on Data Science and Advanced Analytics, DSAA 2018. pp. 80–89 (2019). https://doi.org/10.1109/DSAA.2018.00018
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)

9. Gunning, D., Aha, D.W.: Darpa's explainable artificial intelligence program. AI Magazine **40**(2), 44–58 (2019)
10. Gutierrez, G., Iglesias, J.A., Ordoñez, F.J., Ledezma, A., Sanchis, A.: Agent-based framework for Advanced Driver Assistance Systems in urban environments. In: FUSION 2014 - 17th International Conference on Information Fusion (2014)
11. Hagras, H.: Toward Human-Understandable, Explainable AI. Computer **51**(9), 28–36 (sep 2018). https://doi.org/10.1109/MC.2018.3620965
12. Matheus, C.J., Kokar, M.M., Baclawski, K.: A core ontology for situation awareness. In: Proceedings of the 6th International Conference on Information Fusion, FUSION 2003. vol. 1, pp. 545–552 (2003). https://doi.org/10.1109/ICIF.2003.177494
13. McAree, O., Aitken, J.M., Veres, S.M.: Towards artificial situation awareness by autonomous vehicles. IFAC-PapersOnLine **50**(1), 7038–7043 (jul 2017). https://doi.org/10.1016/j.ifacol.2017.08.1349
14. Meiring, G.A.M., Myburgh, H.C.: A review of intelligent driving style analysis systems and related artificial intelligence algorithms. Sensors (Switzerland) **15**(12), 30653–30682 (2015). https://doi.org/10.3390/s151229822
15. Oulhaci, M.A., Tranvouez, E., Espinasse, B., Fournier, S.: Intelligent tutoring systems and serious game for crisis management: A multi-agents integration architecture. In: Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE. pp. 253–258 (2013). https://doi.org/10.1109/WETICE.2013.78
16. Raptis, D., Iversen, J., Mølbak, T.H., Skov, M.B.: Dara: Assisting drivers to reflect on how they hold the steering wheel. In: ACM International Conference Proceeding Series. pp. 1–12 (2018). https://doi.org/10.1145/3240167.3240186
17. Sipele, O., Zamora, V., Ledezma, A., Sanchis, A.: Advanced Driver's Alarms System through Multi-agent Paradigm. In: 2018 3rd IEEE International Conference on Intelligent Transportation Engineering, ICITE 2018. pp. 269–275 (2018). https://doi.org/10.1109/ICITE.2018.8492600
18. Sukthankar, R., Hancock, J., Pomerleau, D., Thorpe, C.: A Simulation and Design System for Tactical Driving Algorithms. In: Proceedings of AI, Simulation and Planning in High Autonomy Systems (Vol. 6) (1996)
19. Weevers, I., Kuipers, J., Brugman, A.O., Zwiers, J., van Dijk, E.M., Nijholt, A.: The virtual driving instructor creating awareness in a multiagent system. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 2671, pp. 596–602 (2003). https://doi.org/10.1007/3-540-44886-1_56
20. Woolf, B.P.: Advances in Intelligent Tutoring Systems. Studies in Computational Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14363-2
21. Zamora, V., Sipele, O., Ledezma, A., Sanchis, A.: Intelligent agents for supporting driving tasks: An ontology-based alarms system. In: VEHITS 2017 - Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems. pp. 165–172 (2017). https://doi.org/10.5220/0006247601650172
22. Zhao, L., Ichise, R., Yoshikawa, T., Naito, T., Kakinami, T., Sasaki, Y.: Ontology-based decision making on uncontrolled intersections and narrow roads. In: IEEE Intelligent Vehicles Symposium, Proceedings. vol. 2015-Augus, pp. 83–88. IEEE (2015). https://doi.org/10.1109/IVS.2015.7225667