Ontologies for Advanced Driver Assistance Systems

Lihua Zhao[1] Ryutaro Ichise[2] Seiichi Mita[1] Yutaka Sasaki[1] *

[1]Toyota Technological Institute, Japan
[2]National Institute of Informatics, Japan

Abstract: Many *Advanced Driver Assistance Systems* (ADAS) have been developed to improve car safety. A Knowledge Base is indispensable for autonomous vehicles to perceive driving environments and understand traffic regulations. In this paper, we introduce an ontology-based Knowledge Base, which contains maps and traffic regulations. By accessing to the Knowledge Base, the intelligent vehicles can aware overspeed situations and make decisions at intersections in comply with traffic regulations. Two simple ADAS systems are developed based on the Knowledge Base. We conducted field test with an intelligent vehicle to evaluate the ADAS systems.

1 INTRODUCTION

Advanced Driver Assistance Systems (ADAS) can assist autonomous vehicles in safe driving by processing sensor data such as GPS, velocity, and heading angle, etc. Autonomous vehicles should be able to perceive the driving environments and make appropriate decisions according to different traffic situations. For example, slow down when they detect overspeed situations or make decisions at intersections in comply with traffic regulations. To enable the vehicles to understand and to obey traffic regulations, we need a semantic knowledge representation method for the vehicles.

Ontologies are the structural frameworks for organizing information and are used in artificial intelligence, Semantic Web, biomedical informatics, and information architecture as a form of knowledge representation about the world or some part of it. An ontology mainly consists of concepts and the relationships among them. *Resource Description Framework* (RDF) is designed for conceptual description to provide a clear specification for modeling data [3]. In order to represent sensor stream data, we use a timestamp-based temporal RDF representation [5]. Semantic Web developers use *SPARQL Protocol and RDF Query Language* (SPARQL) to access to RDF data [6]. To represent traffic regulations, we use *Semantic Web Rule Language* (SWRL), which is used to express rules as well as logics in Semantic Web applications [1].

Knowledge Base is an indispensable resource for autonomous vehicles to perceive driving environments and to understand traffic regulations. To improve the safety of autonomous vehicles, we construct ontology-based Knowledge Base to assist vehicles to retrieve knowledge and make safe driving decisions. The most cost-efficient way to improve roadway safety is to avoid overspeeding. Furthermore, autonomous vehicles should be able to obey Right-of-Way traffic regulations at intersections. In this paper, we introduce ontologies for constructing Knowledge Base and simple ADAS systems that can improve safety for autonomous vehicles.

*{lihua, smita, yutaka.sasaki}@toyota-ti.ac.jp[1], ichise@nii.ac.jp[2]

The remainder of this paper is organized as follows. In Section 2, we introduce some related research papers. We introduce ontologies for Knowledge Base in Section 3 and introduce two ADAS systems in Section 4. Experimental evaluations are described in Section 5. We conclude our research and propose future work in Section 6.

2 Related Work

Two ontologies about automation levels and situation assessment for *Intelligent Transportation Systems* (ITS) are introduced for co-driving [8]. One ontology defines the relationship between the automation levels and the algorithmic needs. The other ontology is related to the situation assessment level including the concepts of driver, ego-vehicle, communication, free zone, moving obstacles, and environment. Inference rules are defined to link the situation assessment ontology to the automation level ontology, which contains five control levels of a car: longitudinal control, lateral control, local planning, parking, and global planning.

An ontology-based context awareness ADAS system, which utilizes Open Street Map with detailed static entities such as stop intersections and pedestrian crossings is introduced in [4]. The system uses a simple ontology that contains context concepts such as Mobile Entity, Static Entity, and context parameters. The vehicle understands the interactions between the perceived entities and contextual data and performs ontology reasoning with 14 rules written in *Semantic Web Rule Language* (SWRL).

A complex intersection ontology, which contains concepts of car, crossing, road connection (lane and road), and sign at crossing (traffic light and traffic sign) is introduced to form a lean ontology to facilitate fast reasoning [7].

An ontology-based traffic model that can represent typical traffic scenarios such as intersections, multi-lane roads, opposing traffic, and bi-directional lanes is introduced in [9]. Relations such as opposing, conflicting, and neighboring are introduced to represent the semantic context of the traffic scenarios for decision making.

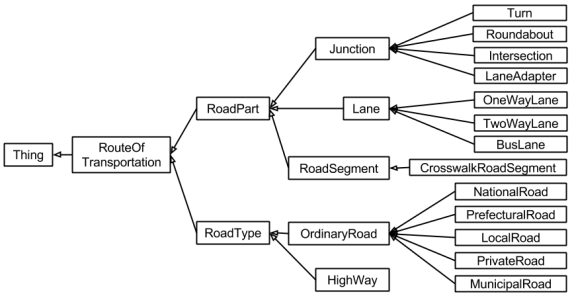

Figure 1: Map ontology.

3 Ontology-Based Knowledge Base

Knowledge Base (KB) is indispensable for autonomous cars to perceive driving environments and to make safe driving decisions. To construct a machine-understandable Knowledge Base, we construct several ontologies, instances, and rules for autonomous cars. The main concepts of ontologies are as follows:

- **Classes** are interpreted as a set of instances in the domain defined by owl:Class. Classes are also called concepts that are the main entities of an ontology.
- **Properties** are also called predicates or relations, which are mainly categorized into owl:ObjectProperty and owl:DatatypeProperty.
- **Instances** are interpreted as particular individuals of a domain, which are defined by owl:Thing.
- **Rules** are statements in the form of an if-then sentence that describe the logical inferences.

3.1 Ontology

We constructed three ontologies for autonomous cars to represent machine-understandable Knowledge Base.

- Map Ontology
  A sophisticated map with road, intersection, lane information is required for autonomous cars to perceive driving environments. Therefore, we construct a map ontology to describe road networks. Figure 1 shows the main classes of the map ontology, where the concepts are linked with rdfs:subClassOf relation. A road consists of connected road parts such as junctions, lanes, and road segments.

  Object properties and datatype properties are used to describe relations among concepts and the value of properties. We constructed some object properties such as map:goStraightTo, map:turnLeftTo, and map:turnRightTo to identify the driving directions between different road parts. The datatype property map:speedMax[1] is used to describe the speed limit of a road and map:boundPOS is used to describe boundary GPS point of junctions and road segments.

[1]PREFIX map: <http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Map#>

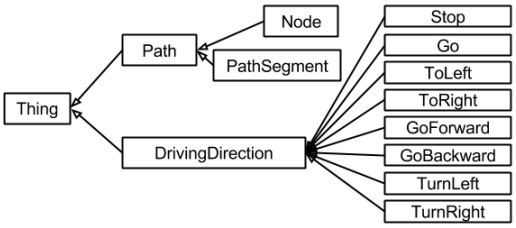

Figure 2: Control ontology.

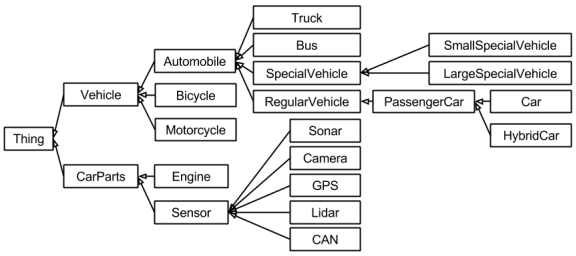

Figure 3: Car Ontology.

- Control Ontology
  The classes shown in Fig. 2 are used to represent paths of autonomous vehicles and driving direction of a vehicle. To represent a path, we use instances of control:PathSegment[2] instead of a collection of GPS points of a trajectory. A path segment can be an intersections, a lane, a crosswalk, or a turn. The Node class contains startNode and endNode, which are the start and end GPS positions of a path. We use the datatype property control:pathSegmentID to index path segments and use the object property control:nextPathSegment to link connected path segments. The object property control:collisionWarningWith is defined to indicate upcoming collisions between our vehicle and the other vehicles.
- Car Ontology
  The car ontology contains concepts of different types of vehicles and devices which are installed on a car such as sensors and engines as shown in Fig. 3. The object property car:isRunningOn[3] is designed to assert the current location of a car. We also added other datatype properties such as car_ID, car_length, and velocity to describe a car.

3.2 Instances

Instances are also known as individuals that model abstract or concrete objects based on the ontologies. With the three ontologies, we model instances such as maps, paths, and cars. Table 1 shows instances in the Knowledge Base near an uncontrolled intersection AnoNagoyaInt3_4. The instances are based on map ontology, which

[2]PREFIX control:<http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Control#>

[3]PREFIX car: <http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Car#>

Table 1: Map ontology based instances.

Subject	Property	Object
AnoNagoyaLine	rdf:type	map:PrefecturalRoad
AnoNagoyaLine	map:hasIntersection	AnoNagoyaInt3_4
AnoNagoyaLine	map:hasRoadSegment	AnoNagoyaRS3
AnoNagoyaLine	map:hasRoadSegment	AnoNagoyaRS4
AnoNagoyaLine	map:speedMax	"40"^^kmh
AnoNagoyaLine	map:osm_way_id	osm_way:122098916
AnoNagoyaInt3_4	rdf:type	map:Intersection
AnoNagoyaInt3_4	map:isConnectedTo	AnoNagoyaRS3
AnoNagoyaInt3_4	map:isConnectedTo	AnoNagoyaRS4
AnoNagoyaInt3_4	map:isConnectedTo	GrandirLaneAdapter1
AnoNagoyaInt3_4	map:boundPos	35.134697, 136.964103
AnoNagoyaInt3_4	map:boundPos	35.134762, 136.964181
AnoNagoyaInt3_4	map:boundPos	35.134788, 136.964072
AnoNagoyaRS4	rdf:type	map:RoadSegment
AnoNagoyaRS4	map:isConnectedTo	AnoNagoyaInt3_4
AnoNagoyaRS4	map:isConnectedTo	AnoNagoyaCrossWalk1
AnoNagoyaRS4	map:boundPos	35.134697, 136.964103
AnoNagoyaRS4	map:boundPos	35.134574, 136.964147
AnoNagoyaRS4Lane2	rdf:type	map:OneWayLane
AnoNagoyaRS4Lane2	map:isLaneOf	AnoNagoyaRS4
AnoNagoyaRS4Lane2	map:enterPos	35.134570, 136.964125
AnoNagoyaRS4Lane2	map:exitPos	35.134693, 136.964082
AnoNagoyaRS4Lane2	control:turnRightTo	GrandirLaneAdapter1
AnoNagoyaRS4Lane2	control:goStraightTo	AnoNagoyaRS3Lane2

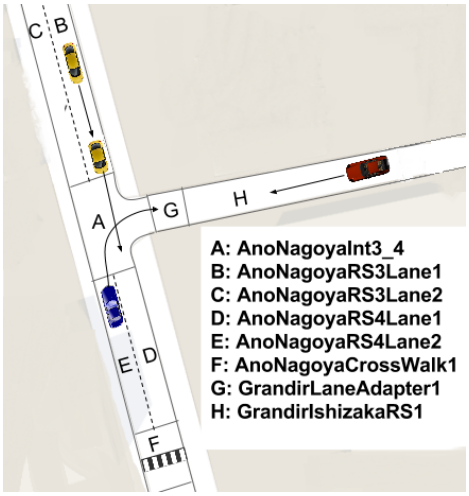

Figure 4: Uncontrolled intersection AnoNagoyaInt3_4.

uses map:hasIntersection or map:hasRoadSegment to link a road with an intersection or a road segment. We use the object property map:isConnectedTo to link intersections with road segments and use map:isLaneOf to relate lanes with road segments. The road AnoNagoyaLine relates to the OpenStreetMap instance osm_way:122098916[4], which has speed limit of 40km/h. In Table 1, the RDF triple $<$ AnoNagoyaRS4Lane2, control:turnRightTo, GrandirLaneAdapter1 $>$ indicates that a car which is currently running on AnoNagoyaRS4Lane2 will run on GrandirLaneAdapter1 if it turns right as shown in Fig. 4.

A vehicle has a path instance, which contains connected path segments and their index numbers. Table 2 shows an example of a path from AnoNagoyaRS4Lane2 to GrandirLaneAdapter1 as shown in Fig. 4. The vehicle updates the next target node position when it changes from one path segment to the next path segment, which is normally the enter or exit position of a lane.

[4]PREFIX osm_way:<http://www.openstreetmap.org/way/>

Table 2: An example of a path instance.

Subject	Property	Object
path:Path1	rdf:type	control:Path
path:Path1	control:startLane	AnoNagoyaRS4Lane2
path:Path1	control:endLane	GrandirLaneAdapter1
AnoNagoyaRS4Lane2	rdf:type	control:StartLane
AnoNagoyaRS4Lane2	control:pathSegmentID	0
AnoNagoyaRS4Lane2	control:nextPathSegment	AnoNagoyaInt3_4
AnoNagoyaInt3_4	control:pathSegmentID	1
AnoNagoyaInt3_4	control:nextPathSegment	GrandirLaneAdapter1
GrandirLaneAdapter1	rdf:type	control:EndLane
GrandirLaneAdapter1	control:pathSegmentID	2

3.3 SWRL Rules

The *Semantic Web Rule Language* (SWRL) is used to express traffic rules. We constructed SWRL rules in the Knowledge Base to handle Right-of-Way rules at uncontrolled intersections. The following two rules are used to make decisions when our vehicle approaches to an uncontrolled intersection and receives a collision warning signal. This Right-of-Way rule means if there is a collision warning, the car which is going to turn right should stop and give way to the other car that is driving straight.

collisionWarningWith(?carX, ?carY)
$\Rightarrow$ CollisionWarning(?carX) $\wedge$ CollisionWarning(?carY)
CollisionWarning(?carX) $\wedge$ CollisionWarning(?carY)
$\wedge$ GoForward(?carY) $\wedge$ TurnRight(?carX)
$\Rightarrow$ Stop(?carX) $\wedge$ giveWay(?carX, ?carY)

The following rule infers a car's driving direction by considering the relations between lanes. We have similar rules for control:GoForward and control:TurnLeft.

isRunningOn(?car, ?lane1) $\wedge$ turnRightTo(?lane1, ?lane2)
$\wedge$ nextPathSegment(?lane1, ?int) $\wedge$ Intersection(?int)
$\wedge$ nextPathSegment(?int, ?lane2) $\Rightarrow$ TurnRight(?car)

SWRL rules in the Knowledge Base are inferred with Pellet reasoner, which provides standard and cutting-edge reasoning services for OWL ontologies [10].

3.4 SPARQL Queries

SPARQL is a powerful RDF query language that enables Semantic Web users to access to the ontology-based Knowledge Base. The following SPARQL query retrieves the next path segment with current path segment and current pathSegmentID "curID". The pathSegmentID starts from 0 and increments by 1. By assigning the pathSegmentID, we can easily identify the next path segment even the current path segment has more than one pathSegmentID.

```
SELECT DISTINCT ?next
WHERE {
  currentPathSegment  control:nextPathSegment  ?next.
  currentPathSegment  control:pathSegmentID  "curID"^^xsd:int.
  ?next  control:pathSegmentID  ?nextID.
  Filter ( ?nextID = (curID + 1) ) }
```

To retrieve the maximum allowed speed of current path segment, we use the following SPARQL query. Here, the

current path segment can be either a road segment or an intersection.

```
SELECT ?max
WHERE {
{ currentPathSegment map:isLaneOf ?roadsegment.
  ?roadsegment map:isRoadSegmentOf ?road.
  ?road map:speedMax ?max.
} UNION {
  ?road map:hasIntersection currentPathSegment.
  ?road map:speedMax ?max. } }
```

When the rule inferencing is performed by the SWRL rule reasoner, we use a SPARQL query to check if we need to give way to the other cars. The following SPARQL query is used to retrieve all the cars that car:ToyotaEstima should give way to when it receives a collision warning.

```
SELECT DISTINCT ?cars
WHERE { car:ToyotaEstima control:giveWay ?cars. }
```

We also constructed other SPARQL queries to retrieve the types of a path segment, the next target node, etc. Jena API[5] is used for executing SPARQL queries on the Knowledge Base.

3.5 C-SPARQL Query

RDF stream data are represented in the format of RDFQuadruple <Subject, Property, Object, Timestamp>. To query on the RDF stream data, we register the OverSpeedCheck as shown in the following C-SPARQL query . The C-SPARQL query checks if a car's average velocity in the past 500ms exceeds its own allowed maximum speed (i.e. 120km/h). The RANGE is the duration to receive sensor stream data for analysis and the STEP size is the frequency of a sensor receiver.

```
REGISTER QUERY OverSpeedCheck AS
SELECT ?car
FROM STREAM
<http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/stream>
  [RANGE 500ms STEP 50ms]
WHERE { ?car car:velocity ?speed . }
GROUP BY ?speed
HAVING (AVG(?speed) >= maxSpeed )
```

4 ADAS Systems

We constructed an ontology-based Knowledge Base to assist drivers or autonomous vehicles to drive safely on urban roads. In this section, we introduce two ADAS systems that utilize the ontology-based Knowledge Base: an Intelligent Speed Adaptation system that can detect overspeed situations and a decision making system that can make driving decisions at uncontrolled intersections.

[5] http://jena.apache.org/documentation/ontology/

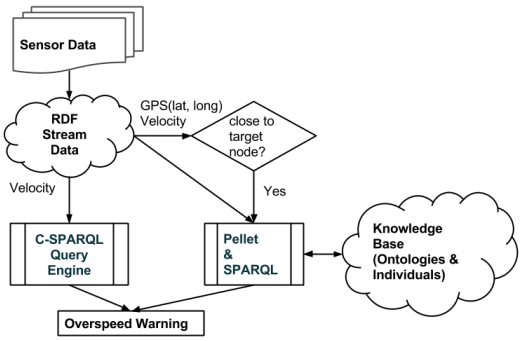

Figure 5: Flowchart of ISA system.

4.1 Intelligent Speed Adaptation System

An Intelligent Speed Adaptation (ISA) system enables the autonomous cars to perceive driving environment and detect overspeed situations. The ISA system retrieves knowledge of driving environments such as speed limit information from the ontology-based Knowledge Base.

Figure 5 shows the flowchart of the ISA system. The sensor data are converted into RDF stream data to detect overspeed. With C-SPARQL query, we check if a car's velocity exceeds its own maximum speed limit. We execute SPARQL queries on the inferred Knowledge Base to retrieve current lane, speed limit, and update the next target node according to the received real-time sensor data. SPARQL queries are only performed when the car approaches to the closest position from the current target node. By comparing the velocity with the speed limit retrieved from the Knowledge Base, we can send overspeed warning to the autonomous car.

4.2 Intelligent Decision Making System

Figure 6 shows the flowchart of the decision making system, which accesses to the ontology-based Knowledge Base to make driving decisions such as "Stop", "ToLeft", or "Give Way", etc. The decision making system mainly consists of a sensor data receiver, an ontology-based Knowledge Base, a SPARQL query engine, and a SWRL rule reasoner. The main processing steps of the decision making system are as follows:

1. The sensor data receiver sends received collision warning signal and the other vehicle's information to the SPARQL query engine and SWRL rule reasoner.
2. The SPARQL query engine accesses to the Knowledge Base to retrieve information of our vehicle's current lane, next lane, and driving direction, etc..
3. SWRL rule reasoner adds additional information such as collision warning and the other vehicle's position, velocity, and driving direction to the Knowledge Base. For example, if we detect our car (carX) has a collision warning with carY, we add a triple <carX, control:collisionWarningWith, carY>.
4. SWRL rule reasoner performs reasoning on the updated Knowledge Base and new inferred information is added to the Knowledge Base. For example,

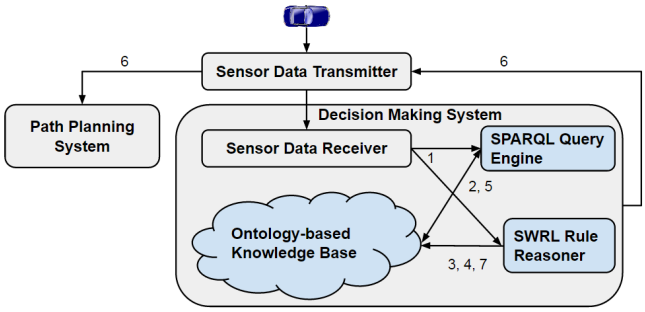

Figure 6: Flowchart of decision making system.

(a) Experimental path (b) Experimental result

Figure 7: Experimental result of ISA system.

decisions such as “Stop”, “ToLeft”, or ”Give Way” with the other vehicle’s ID.

5. The SPARQL query engine accesses to the Knowledge Base to retrieve the commands and the vehicles that our vehicle should give way to.

6. The decision signals are sent to the path planning system via the sensor data transmitter to update driving path or driving behavior.

7. Newly added inferred knowledge is removed from the ontology-based Knowledge Base.

5 Experiments

In this section, we evaluate the ISA system and decision making system using real-time sensor data from an Estima car while it drives on the predefined driving path.

5.1 Data Format

Table 3 shows the format of sensor data transmitted through User Datagram Protocol (UDP) [2] at real-time. At each timestamp, the sensor data transmitter sends data in the order of timestamp, latitude, longitude, velocity, heading angle, carID, and collision warning signal. Here, if the collision warning signal is 0, it means that no upcoming collision is detected. Otherwise, it means that an upcoming collision is detected and sends the detected vehicle’s information along with our experimental vehicle’s information. The default ID for our experimental vehicle is 0 and the other detected vehicle’s IDs are non-zero integers.

5.2 Evaluation of ISA System

We collected sensor data by running on the same path as shown in Fig. 7a, which took about 13 minutes. The ISA system is tested on the collected sensor data to detect overspeed situations.

Figure 7b shows the experimental result with the real-world sensor data. The cross marks represent overspeed positions and the cross marks in the circle represent overspeed positions near the Takasaka kindergarten. The experimental result shows that even an experienced driver may drive faster than the speed limit, which is normally shown with a road sign on the road. The running time for reasoning with Pellet reasoner is 177ms and the execution time for three SPARQL queries is 11ms. The maximum query time is 23ms and minimum 3ms.

With the ontology-based Knowledge Base and precise GPS positions from the GPS-IMU sensor, we can promptly detect the current running lane or intersection and retrieve the speed limit information. The accuracy of the overspeed detection may be affected by the delay of data transmission from the GPS-IMU sensor or by the execution time of three SPARQL queries. Therefore, the ideal execution time for overspeed detection should be less than the frequency of the GPS-IMU sensor, which is 5ms in our experiment.

5.3 Evaluation of Decision Making System

To evaluate whether the decision making system can make correct driving decisions at real-time, we conducted experiments with the intelligent vehicle (Toyota Estima) on the driving path that contains road segments of AnoNagoyaRS4Lane2, AnoNagoyaInt3_4, and GrandirLaneAdapter1 as introduced in Table 2. We chose this kind of area in Nagoya city of Japan because it contains an uncontrolled intersection, which is very common in Japan. Furthermore, it’s a challenging task for autonomous vehicles to drive safely at uncontrolled intersections. The intelligent vehicle is equipped with many sensors such as Velodyne Lidar, GPS-IMU, and cameras.

The sensor data transmitter sends sensor data in the format as shown in Table 3 while the intelligent vehicle turns right at an uncontrolled intersection as shown in Fig. 4. Table 4 shows the decisions made in different timestamps according to the transmitted sensor data in Table 3. The decision making system is executed only when the vehicle receives a collision warning signal. From timestamp 1422322 until 1424945, our vehicle stops and gives way to the other vehicles which were running straight from AnoNagoyaRS3Lane1 to the intersection AnoNagoyaInt3_4. At timestamp 1425039 the collision warning is cleared and our vehicle can move and turn right to the narrow road. The average execution time for making a decision is about 99ms, which ranges from 79ms to 312ms.

Table 3: An example of transmitted sensor data

Timestamp	Latitude	Longitude	Velocity (m/s)	Heading Angle	Car ID	Collision Warning
1422228	35.134644	136.964111	0.103047	-345.415405	0	0
1422322	35.134644	136.964111	0.092203	-345.413116	0	1
1422322	35.134892	136.964081	9.589874	189.68808	1	1
...	...	...	...	...	...	...
1424945	35.134647	136.964111	0.216733	-345.326355	0	1
1424945	35.134682	136.964127	9.596083	200.214127	1	1
1425039	35.134647	136.964111	0.210857	-345.324646	0	0

Table 4: Experimental result of decision making system.

Timestamp	Estima Position	Detected Vehicle	Decision
1422228	AnoNagoyaRS4Lane2	N/A	Go
1422322	AnoNagoyaRS4Lane2	AnoNagoyaRS3Lane1	Stop, Give Way
...	...	...	...
1424945	AnoNagoyaRS4Lane2	AnoNagoyaInt3_4	Stop, Give Way
1425039	AnoNagoyaRS4Lane2	N/A	Go

To infer the Right-of-Way rules at uncontrolled intersections, we need the information of the other vehicle's driving direction. However, it is still a challenging problem to identify if the detected vehicle is going to turn left or turn right when they approach an intersection. The Collision Detection system used in our experiment cannot retrieve the intended driving direction of the other vehicles by observing the head lights of the other vehicles. Therefore, we assume that we can observe the other vehicle's intended driving direction as human drivers do with predefined driving direction.

6 Conclusion and Future Work

Driving on uncontrolled intersections is a challenging problem and is common in urban areas of Japan. In this paper, we proposed an ontology-based Knowledge Base and two ADAS systems. The Intelligent Speed Adaptation system detects overspeed situations and the intelligent decision making system can make safe driving decisions at uncontrolled intersections. The Knowledge Base contains instances of roads and Right-of-Way rules written in SWRL. We chose some areas of Nagoya to evaluate the systems with real-time sensor data. Experimental results show that the ISA system can promptly detect overspeed situations and the decision making system can effectively make decisions such as "Stop", "ToLeft", or "Give Way".

In future work, we will speed up the reasoning time for making a decision by using part of Knowledge Base. Furthermore, we will cover other possible situations such as at the corner or in private roads near apartments.

References

[1] *SWRL: A Semantic Web Rule Language Combining OWL and RuleML.* `http://www.w3.org/Submission/SWRL/`.

[2] *UDP - User Datagram Protocol.* `http://ipv6.com/articles/general/User-Datagram-Protocol.htm`.

[3] Dean Allemang and James A. Hendler. *Semantic Web for the Working Ontologist - Effective Modeling in RDFS and OWL.* Morgan Kaufmann, second edition, 2011.

[4] Alexandre Armand, David Filliat, and Javier Ibañez-Guzman. Ontology-Based Context Awareness for Driving Assistance Systems. In *IEEE Intelligent Vehicles Symposium*, pages 227–233, 2014.

[5] Claudio Gutierrez, Carlos A. Hurtado, and Alejandro Vaisman. Introducing Time into RDF. *IEEE Transactions On Knowledge and Data Engineering*, 19(2):207–218, 2007.

[6] Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 2011.

[7] Michael Hülsen, J. Marius Zöllner, and Christian Weiss. Traffic Intersection Situation Description Ontology for Advanced Driver Assistance. In *IEEE Intelligent Vehicles Symposium*, pages 993–999. IEEE, 2011.

[8] Evangeline Pollard, Philippe Morignot, and Fawzi Nashashibi. An Ontology-Based Model to Determine the Automation Level of An Automated Vehicle for Co-Driving. In *16th International Conference on Information Fusion*, pages 596–603, 2013.

[9] Ralf Regele. Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles. In *4th International Conference on Autonomic and Autonomous Systems*, pages 94–99. IEEE Computer Society, 2008.

[10] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, 2007.

[11] Lihua Zhao, Ryutaro Ichise, Seiichi Mita, and Yutaka Sasaki. An Ontology-Based Intelligent Speed Adaptation System for Autonomous Cars. In *4th Joint International Semantic Technology Conference*. Springer Berlin Heidelberg, 2014. (to appear).