2015 IEEE Intelligent Vehicles Symposium (IV)
June 28 - July 1, 2015. COEX, Seoul, Korea

Ontology-based Decision Making on Uncontrolled Intersections and Narrow Roads

Lihua Zhao[1], Ryutaro Ichise[2], Tatsuya Yoshikawa[3], Takeshi Naito[3], Toshiaki Kakinami[3], and Yutaka Sasaki[1]

***Abstract*— Many *Advanced Driver Assistance Systems* (ADAS) have been developed to improve car safety. However, it is still a challenging problem to make autonomous vehicles to drive safely on urban streets such as uncontrolled intersections (without traffic lights) and narrow roads. In this paper, we introduce a decision making system that can assist autonomous vehicles at uncontrolled intersections and narrow roads. We constructed a machine understandable ontology-based Knowledge Base, which contains maps and traffic regulations. The system makes decisions in comply with traffic regulations such as Right-Of-Way rules when it receives a collision warning signal. The decisions are sent to a path planning system to change the route or stop to avoid collisions.**

I. INTRODUCTION

Autonomous vehicles should be able to perceive the driving environments and make appropriate decisions according to different traffic situations. *Advanced Driver Assistance Systems* (ADAS) can assist autonomous vehicles to make decisions by processing sensor data such as GPS, velocity, and heading angle, etc. To enable the vehicles to understand sensor data and traffic regulations, we need a semantic knowledge representation method.

Ontologies can represent knowledge of things for the ADAS systems in a machine-readable format with concepts and the relationships among them. *Resource Description Framework* (RDF) is designed for conceptual description to provide a clear specification for modeling data [1]. We use a timestamp-based temporal RDF representation to represent the stream data from the sensors [2]. *SPARQL Protocol and RDF Query Language* (SPARQL) is a powerful RDF query language that enables Semantic Web users to access to static RDF data [3]. The *Semantic Web Rule Language* (SWRL) is a used to express rules as well as logics in Semantic Web applications [4].

Currently, many autonomous vehicles can run on controlled intersections or on highways. However, running on urban streets such as uncontrolled intersections and narrow roads still remains as a challenging problem. In Japan, there are many narrow roads where even human drivers feel difficulty in driving. As shown in Fig. 1, when a car approaches an uncontrolled intersection that has no traffic lights, the driver has to carefully observe the other vehicles to decide whether to give way or not. The lane adapter (GrandirLaneAdapter1) connected to the intersection only allows one vehicle to drive on and the narrow road (GrandirIshizakaRS1) is extremely difficult for two vehicles to run freely. It's safer to stop on the left side and give way to the other vehicle to pass by slowly. Therefore, we developed a decision making system that can make safety decisions on uncontrolled intersections and narrow two-way roads by accessing to an ontology-based Knowledge Base.

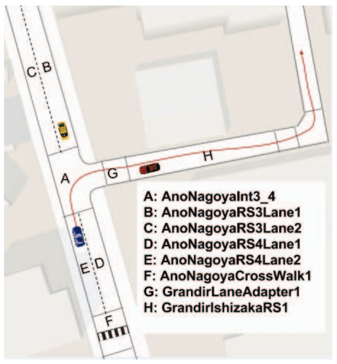


Fig. 1: Uncontrolled intersections and narrow roads.

The remainder of this paper is organized as follows. In Section II, we introduce related research papers and describe the advantages of our research. In Section III, we introduce the ontology-based decision making system in detail. We evaluate the decision making system in Section IV using simulation data and real-world data. We conclude our research and propose future work in Section V.

II. Related Work

An ontology-based context awareness ADAS system is introduced in [5], which understands the interactions between the perceived entities and contextual data. A simple ontology that includes context concepts such as Mobile Entity, Static Entity, and context parameters is modeled to enable the vehicle to understand the context information when it approaches road intersections. By applying 14 rules written in *Semantic Web Rule Language* (SWRL), the ontology is able to process human-like reasoning on global road contexts.

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 that is close to real-time [6]. Another ontology-based traffic model that can represent typical traffic scenarios such as intersections,

[1]Research Center for Smart Vehicles, Toyota Technological Institute, 2-12-1 Hisakata, Tempaku, Nagoya, Aichi 468-8511, Japan {lihua, yutaka.sasaki}@toyota-ti.ac.jp
[2]National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan ichise@nii.ac.jp
[3] Chassis & Vehicle System Development Dept., AISIN SEIKI Co., Ltd., 2-1 Asahimachi, Kariya, Aichi 448-8650, Japan yoshi@its.aisin.co.jp,t-naito@rd.aisin.co.jp, kakinami@rd.aisin.co.jp

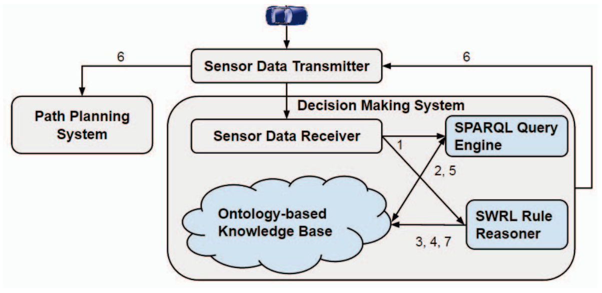


Fig. 2: Flowchart of decision making system.

multi-lane roads, opposing traffic, and bi-directional lanes is introduced in [7]. Relations such as opposing, conflicting, and neighboring are introduced to represent the semantic context of the traffic scenarios for decision making.

An ontology-based Knowledge Base for Intelligent Speed Adaptation system is introduced in [8], which processes stream data from GPS-IMU sensor to detect overspeed situations. The ontology-based Knowledge Base contains maps, paths, and SWRL rules, which are based on three ontologies: map, control, and car ontology. The system can promptly detect overspeed situations by accessing to the ontology-based Knowledge Base.

In contrast to above research, we focus on narrow road cases and the advantages are:

- With a path and an ontology-based map, we can retrieve driving direction of the vehicle at intersections and semantic road information at real-time.
- Right-Of-Way rules written in SWRL are used for rule inference to make safe driving decisions when the vehicle receives collision warning signals.
- We focus on uncontrolled intersections and narrow two-way roads, which are more difficult to make decisions than running on controlled intersections or highways.

III. Approach

Figure 2 shows the flowchart of decision making system, which accesses to the ontology-based Knowledge Base to make driving decisions such as “Stop”, “ToLeft”, or ”Give Way”, etc. 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. For example, in Fig. 1, we retrieve that our vehicle is going to run on the intersection AnoNagoyaInt3_4 and its driving direction is “TurnRight”.
3) SWRL rule reasoner adds some additional information such as collision warning and the other vehicle’s position, velocity, and driving direction into the Knowledge Base. For example, if we detected that our vehicle (carX) has a collision warning with carY, we add a triple <carX, control:collisionWarningWith[1], carY>.
4) SWRL rule reasoner performs reasoning on the updated Knowledge Base and new inferred information is added to the Knowledge Base. For example, 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.

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. In the following, we describe each component in detail.

A. Knowledge Base

The ontology-based Knowledge Base used in this paper is an extension of the research [8]. For practical usage of ontologies, they should support large-scale interoperability, to be well-founded and axiomatized to be generally understood [9]. Ontologies are expressed in an ontology language, which usually deal with the following kinds of entities [10]:

- **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.

Protégé ontology editor is an open development environment for Semantic Web applications [11]. In the following, we describe the main extensions of ontologies that are used for the decision making system.

1) Ontology:
We extended the map and control ontology of previous research [8] to enable the autonomous vehicles to understand traffic situations on intersections and narrow roads.

- Map Ontology
  A map is an essential knowledge resource for autonomous vehicles to retrieve driving environment information. Therefore, we constructed a map ontology that can describe different types of roads, intersections, lanes, and the relations among them.
  Figure 3 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. We use some object

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

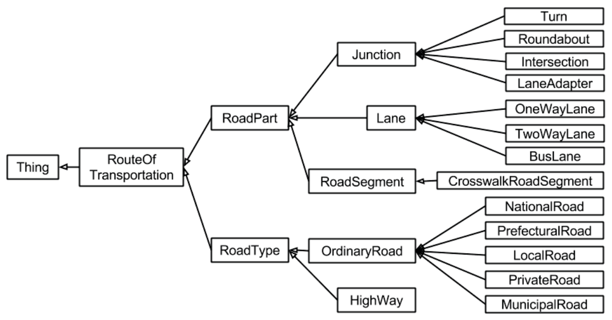


Fig. 3: Map ontology.

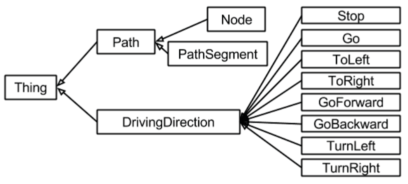


Fig. 4: Control ontology.

TABLE I: Map ontology based instances.

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

properties such as map:goStraightTo, map:turnLeftTo, and map:turnRightTo to assert the relations between lanes as well as to identify the driving direction.

- Control Ontology
  The classes shown in Fig. 4 are used to represent paths and driving directions of autonomous vehicles. To represent a path, we use instances of control:pathSegment instead of a collection of GPS points of a trajectory. A path segment can be any part of roads such as intersections, lanes, crosswalks, or turns. The Node class contains startNode and endNode, which are the start and end GPS positions. We use a datatype property control:pathSegmentID to index path segments and use object property control:nextPathSegment to link connected path segments. An object property control:collisionWarningWith is defined to indicate upcoming collisions among our vehicle and the other vehicles.

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. In the following, we give some examples of map instances in the Knowledge Base including roads, road segments, intersections, lanes, and lane adapters, etc.

Table I shows some map instances in the Knowledge Base such as a prefectural road yagoto:AnoNagoyaLine[2], an uncontrolled intersection yagoto:AnoNagoyaInt3_4, a road segment yagoto:AnoNagoyaRS4 connected to the intersection yagoto:AnoNagoyaInt3_4, and a one-way lane yagoto:AnoNagoyaRS4Lane2 which is a lane of yagoto:AnoNagoyaRS4. The instances are based on map ontology, which uses map:hasIntersection or map:hasRoadSegment to link a road with an intersection or a road segment. The road yagoto:AnoNagoyaLine relates to the OpenStreetMap instance osm_way:122098916[3], which has speed limit of 40km/h. In Table I, the RDF triple <yagoto:AnoNagoyaRS4Lane2, control:turnRightTo, yagoto:GrandirLaneAdapter1> indicates that a car running on yagoto:AnoNagoyaRS4Lane2 is going to turn right to run on yagoto:GrandirLaneAdapter1 as shown in Fig. 1.

A vehicle has a path instance, which contains connected path segments and their index numbers. 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.

3) SWRL Rules:

The *Semantic Web Rule Language* (SWRL) is used to express Right-of-Way rules. We constructed 13 SWRL rules in the Knowledge Base to handle Right-of-Way rules at uncontrolled intersections and on narrow two-way roads.

Table II lists some of the SWRL rules in our Knowledge Base. In rule 1, if carX receives a collision warning with carY, we alert both cars and assert that they both have a collision warning. The second rule in Table II infers carX's driving direction as control:TurnRight by considering the relations between lanes. We have similar rules for control:GoForward and control:TurnLeft.

Rule 3, 4, and 5 are Right-of-Way rules before an uncontrolled intersection, at an uncontrolled intersection, and on a two-way lane, respectively. Rule 3 means that 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. Rule 4 means if the car receives a collision warning when it is running on an intersection, it stops and gives way to the other car. For the safety, we always stop first and then analyze if the other car is also waiting for our vehicle or not. Rule 5 shows that if our car (carX) is on a two-way lane and receives an upcoming collision warning, it should always move to the left side and give way to the other car.

[2]PREFIX yagoto:<http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/YagotoMap#> [3]PREFIX osm_way:<http://www.openstreetmap.org/way/>

TABLE II: Examples of SWRL rules.

1	collisionWarningWith(?carX, ?carY) $\Rightarrow$ CollisionWarning(?carX) $\wedge$ CollisionWarning(?carY)
2	Intersection(?int) $\wedge$ isRunningOn(?carX, ?lane1) $\wedge$ turnRightTo(?lane1, ?lane2) $\wedge$ nextPathSegment(?lane1, ?int) $\wedge$ nextPathSegment(?int, ?lane2) $\Rightarrow$ TurnRight(?carX)
3	CollisionWarning(?carX) $\wedge$ CollisionWarning(?carY) $\wedge$ GoForward(?carY) $\wedge$ TurnRight(?carX) $\Rightarrow$ Stop(?carX) $\wedge$ giveWay(?carX, ?carY)
4	MyCar(?car1) $\wedge$ isRunningOn(?car1, ?int) $\wedge$ Intersection(?int) $\wedge$ collisionWarningWith(?car1, ?car2) $\Rightarrow$ Stop(?car1) $\wedge$ giveWay(?car1, ?car2)

5	TwoWayLane(?lane) $\wedge$ isRunningOn(?carX, ?lane)
	 CollisionWarning(?carX) $\wedge$ CollisionWarning(?carY)
	 $\Rightarrow$ ToLeft(?carX), giveWay(?carX, ?carY)

B. SWRL Rule Reasoner

When the autonomous vehicle receives a collision warning signal from a collision detection system, the SWRL rule reasoner is executed according to different types of traffic situations. Since we mainly focus on uncontrolled intersection and narrow road cases, we categorize the traffic situations into three different types:

- Before an intersection: Give way or move forward in comply with Right-of-Way rules.
- At an intersection: Stop and give way to the other cars when upcoming collisions are detected. If the other cars have been stopped for a specific duration, our car can continue going.
- On a two-way lane: Move to the left side and give way to the other cars coming from the opposite side of the two-way lane.

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. SWRL rules in the Knowledge Base are inferred with Pellet reasoner, which provides standard and cutting-edge reasoning services for OWL ontologies [12]. Pellet API[4] and OWL API[5] are applied for reasoning.

C. SPARQL Query Engine

SPARQL is a powerful RDF query language that enables Semantic Web users to access to the ontology-based Knowledge Base. The following SPARQL query is used to retrieve all the cars that car:ToyotaEstima should give way to when it receives a collision warning signal. The triple including the object property control:giveWay is added to the Knowledge Base when the decision making system infers according to the Right-of-Way rules in SWRL.

```
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, and the speed limit of current path segment, etc. Jena API[6] is used for executing SPARQL queries on the Knowledge Base.

[4]http://clarkparsia.com/pellet/
[5]http://owlapi.sourceforge.net/
[6]http://jena.apache.org/documentation/ontology/

IV. Experiment

In this section, we introduce the sensor data that are transmitted through User Datagram Protocol (UDP) [13] at real-time. The experiments are conducted in two ways. First, we test by simulating different traffic situations to test if the system can make correct decisions. Then, we test with the real-time sensor data sent from the sensor data transmitter by driving an Estima car on the driving path.

A. Experiment Settings

1) Computer Specification:

The computer specifications are as follows.

Experiment	Operating System	CPU (64 bits)	RAM
Simulation Test	Win7 Professional	i7 @3.40GHz	16GB
Real-World Data Test	Win7 Professional	i7 @3.40GHz	32GB

2) Experimental Area:

We chose one part of the Yagoto area in Nagoya city of Japan for our experiment, which contains both an uncontrolled intersection and a narrow two-way lane as shown in Fig. 1. A lane adapter (yagoto:GrandirLaneAdapter1) connects an uncontrolled intersection (yagoto:AnoNagoyaInt3_4) and a two-way narrow road (yagoto:GrandirIshizakaRS1). This kind of roads are very common in Japan and it's a challenging task for autonomous vehicles to drive safely because:

- Since there are no traffic lights at the uncontrolled intersection, the vehicle has to observe the other vehicles and make a decision to give way or not. Therefore, additional knowledge of Right-of-Way traffic regulations at uncontrolled intersections are necessary.
- Even human drivers are cautious when running on two-way narrow roads as shown in Fig. 1. The vehicle should understand that although the road allows cars to run in both directions, the limited space is difficult for two cars drive freely.

B. Data

1) Data Format:

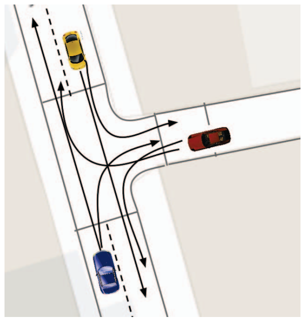
Table III shows the format of sensor data transmitted at real-time to our decision making system. At each timestamp, the sensor data transmitter sends data in the order of timestamp, latitude, longitude, velocity, heading angle, carID, and collision warning signal. Here, if the collision warning signal 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.

2) Knowledge Base for Experiments:

The instances in the Knowledge Base for experiments are based on the map ontology and control ontology. Vehicle instances are created based on the car ontology as introduced in [8] and our experimental vehicle is assigned a path. The instances of other vehicles that have collision warnings with our experimental vehicle are added to the Knowledge Base at real-time and then deleted from the Knowledge Base when the warning signals are cleared.

TABLE III: 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


(a) Uncontrolled intersection

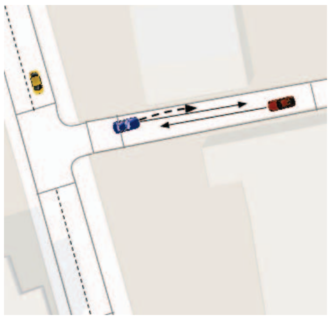

(b) Narrow two-way lane.

Fig. 5: Possible paths for simulation experiment.

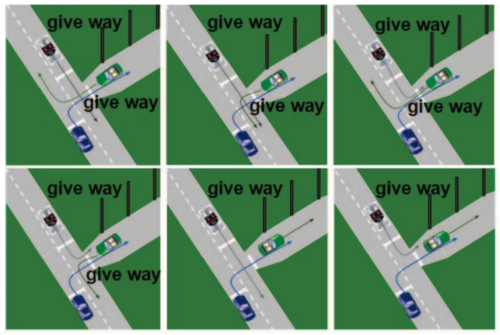


Fig. 6: Experimental results for a path with 3 vehicles.

C. Simulation Experiment

To evaluate the ontology-based decision making system in the experimental area as shown in Fig. 1, we simulated all the possible traffic situations that would happen at the uncontrolled intersection and on the narrow two-way road.

- Paths: As shown in Fig. 5a, there are six possible paths for a vehicle when it approaches the uncontrolled intersection. When a car is running on the two-way lane, there are two possible paths as shown in Fig. 5b.
- Vehicles: When our vehicle approaches the intersection as shown in Fig. 5a, the maximum number of other vehicles that may have upcoming collision with our vehicle is two. We only consider the nearest vehicles on each lane or intersection and the other following vehicles are neglected because the vehicles will be considered when they become the nearest to our vehicle.

To infer the Right-of-Way rules at uncontrolled intersections and on narrow two-way roads, we need the information of the other vehicle's driving direction. It is easy to identify driving direction on a narrow road by considering heading angle, which can be either the same direction or opposing 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.

In total, we tested 24 two-vehicle cases and 32 three-vehicle cases with 6 possible paths to evaluate our decision making system at the uncontrolled intersection. Figure 6 shows the three-vehicle cases for the path of a car which turns right at the intersection and drives on the narrow two-way road. The "give way" labels beside the other vehicles means that our vehicle should give way to the car which may have upcoming collision with our vehicle.

For example, in the first case of Fig. 6, our vehicle should give way to both cars because one vehicle is driving straight and the other vehicle is driving out from narrow road to wide road. According to the Right-of-Way rules, a car driving straight has a higher priority than a car turning right or left at an uncontrolled intersection. If a car is going to drive into a narrow two-way road, it should wait for the car which is going to drive out from the narrow two-way road. In the last two cases of Fig. 6, our vehicle does not need to give way to the car running on the narrow two-way lane, because the driving direction will be the same. Furthermore, if two cars are going to enter the same lane, the car turning left has a higher priority than the car turning right.

On the narrow two-way road, our vehicle gives way to the other vehicle coming from the opposing direction and moves to the left side of the road as the dotted path shown in Fig. 5b. This decision is made according to Rule 1 and Rule 5 in Table II. The "ToLeft" signal is sent to the path planning system to change the current path by finding a position on the left side to stop and to give way to the other vehicle.

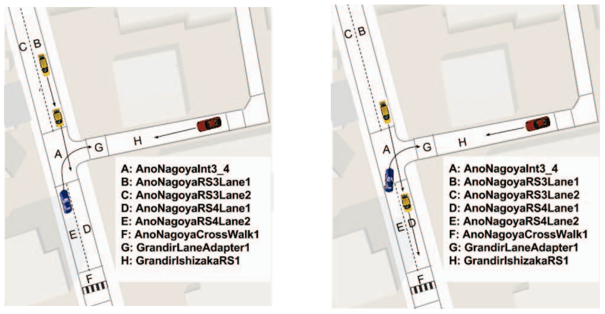
D. Real-World Data Experiment

To evaluate whether the decision making system can make correct decisions at real-time, we tested with the intelligent vehicle (Toyota Estima) on the same driving path. 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 III while the intelligent vehicle runs on the path as shown in Fig. 7a. The path segments for the experiments are AnoNagoyaRS4Lane2, AnoNagoyaInt3_4,

TABLE IV: Experimental results with real-world data.

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



(a) Near AnoNagoyaInt3_4. (b) On AnoNagoyaInt3_4.

Fig. 7: Encountered situations during real-world experiment.

GrandirLaneAdapter1, GrandirIshizakaRS1, and so on. Table IV shows the decisions made in different timestamps according to the data in Table III. The decision making system is executed only when the vehicle receives a collision warning. From time 1422322 until 1424945, our vehicle stops and gives way to the other vehicles. Here, we assume that the detected vehicles run straight from AnoNagoyaRS3Lane1 to AnoNagoyaInt3_4. At time 1425039 the collision warning is cleared and our vehicle can move and turn right to the narrow road.

We drove on the same path for several times and classified the encountered situations when collisions are detected:

- Case 1: As shown in Fig. 7a, the intelligent vehicle is running on AnoNagoyaRS4Lane2 and is going to run on the uncontrolled intersection AnoNagoyaInt3_4. Another vehicle is running straight from AnoNagoyaRS3Lane1 to AnoNagoyaInt3_4.
  Decision: Stop and give way to the other vehicle.
  Average execution time: 99ms (79ms $\sim$ 312ms)
- Case 2: As shown in Fig. 7b, the intelligent vehicle is running on AnoNagoyaInt3_4, and the other vehicle is running straight on AnoNagoyaRS3Lane1.
  Decision: Stop and observe the other vehicle. (Our vehicle can move if the other vehicle stopped for 500ms.)
  Average execution time: 187ms (110ms $\sim$ 327ms)
- Case 3: The intelligent vehicle detected another vehicle coming from the opposite side of the two-way road GrandirIshizakaRS1 as shown in Fig. 5b.
  Decision: Move to the left side of the road and give way to the other vehicle.
  Average execution time: 138ms (84ms $\sim$ 173ms)

As the experimental results shown above, the decision making system makes correct decisions in three different cases for the uncontrolled intersection cases and two-way narrow road cases. However, the execution time for making a decision still needs to be improved. Most of the execution time exceeds 100ms, which means the intelligent vehicle may advance about 1m $\sim$ 2m during the time.

V. Conclusion and Future Work

Driving on uncontrolled intersections and narrow roads is a challenging problem and is common in urban areas of Japan. In this paper, we proposed an ontology-based Knowledge Base and a decision making system that can make safe driving decisions at uncontrolled intersections and narrow two-way roads. The Knowledge Base contains instances of roads and Right-of-Way rules written in SWRL. We chose the Yagoto area of Nagoya for preliminary experiment and performed simulation tests to cover all the possible situations in the specific area. Experimental results show that the decision making system can effectively make decisions such as “Stop”, “ToLeft”, or ”Give Way”.

In future work, we will speed up the processing time for making a decision and include other areas that have uncontrolled intersections and narrow roads. Furthermore, we will cover other possible situations such as at the corner or in private roads near apartments.

Acknowledgment

We would like to express our gratitude to Prof. Seiichi Mita, Dr. Keisuke Yoneda, and Dr. Duoc Huy Do for their assistance and technical support.

References

[1] D. Allemang and J. A. Hendler, *Semantic Web for the Working Ontologist - Effective Modeling in RDFS and OWL*, 2nd ed. Morgan Kaufmann, 2011.

[2] C. Gutierrez, C. A. Hurtado, and A. Vaisman, “Introducing Time into RDF,” *IEEE Transactions On Knowledge and Data Engineering*, vol. 19, no. 2, pp. 207–218, 2007.

[3] T. Heath and C. Bizer, *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 2011.

[4] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*, 2004, http://www.w3.org/Submission/SWRL/.

[5] A. Armand, D. Filliat, and J. Ibañez-Guzman, “Ontology-Based Context Awareness for Driving Assistance Systems,” in *IEEE Intelligent Vehicles Symposium*, 2014, pp. 227–233.

[6] M. Hülsen, J. M. Zöllner, and C. Weiss, “Traffic Intersection Situation Description Ontology for Advanced Driver Assistance,” in *IEEE Intelligent Vehicles Symposium*, 2011, pp. 993–999.

[7] R. Regele, “Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles,” in *4th International Conference on Autonomic and Autonomous Systems*. IEEE Computer Society, 2008, pp. 94–99.

[8] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, “An Ontology-Based Intelligent Speed Adaptation System for Autonomous Cars,” in *4th Joint International Semantic Technology Conference*. Springer Berlin Heidelberg, 2014, pp. 397–413.

[9] S. Staab and R. Studer, *Handbook on Ontologies*, 2nd ed. Springer Berlin Heidelberg, 2009.

[10] J. Euzenat and P. Shvaiko, *Ontology Matching*. Springer Berlin Heidelberg, 2007.

[11] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen, “The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications,” in *3rd International Semantic Web Conference*. Springer, 2004, pp. 229–243.

[12] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, “Pellet: A Practical OWL-DL Reasoner,” *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp. 51–53, 2007.

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