PAPER

Ontology-Based Driving Decision Making: A Feasibility Study at Uncontrolled Intersections

Lihua ZHAO[†a)], ***Nonmember***, **Ryutaro ICHISE**[††b)], ***Member***, **Zheng LIU**[†††c)], ***Nonmember***, **Seiichi MITA**[††††d)], ***Member***, ***and*** **Yutaka SASAKI**[††††e)], ***Nonmember***

SUMMARY This paper presents an ontology-based driving decision making system, which can promptly make safety decisions in real-world driving. Analyzing sensor data for improving autonomous driving safety has become one of the most promising issues in the autonomous vehicles research field. However, representing the sensor data in a machine understandable format for further knowledge processing still remains a challenging problem. In this paper, we introduce ontologies designed for autonomous vehicles and ontology-based knowledge base, which are used for representing knowledge of maps, driving paths, and perceived driving environments. *Advanced Driver Assistance Systems* (ADAS) are developed to improve safety of autonomous vehicles by accessing to the ontology-based knowledge base. The ontologies can be reused and extended for constructing knowledge base for autonomous vehicles as well as for implementing different types of ADAS such as decision making system.

key words: *ontology, knowledge representation, knowledge base, SPARQL, C-SPARQL, autonomous vehicles, decision making systems, Advanced Driver Assistance System (ADAS)*

1. Introduction

Autonomous driving is one of the most promising and challenging research topics in the automobile industries as well as the IT industries, and academic research centers. Current autonomous vehicles under development are equipped with varied sensors such as mono camera, stereo camera, Lidar, and Radar. By analyzing sensor data, the autonomous vehicles can detect lanes for lane keeping or avoid accidents by detecting pedestrians and other obstacles. Although objects and lanes can be detected using these sensors, the vehicles cannot understand the meanings of driving environments without knowledge representation of the data. Therefore, a machine understandable knowledge representation method is critical to fill the gap between driving environments perception and further knowledge processing.

Manuscript received August 10, 2016.
Manuscript revised February 21, 2017.
Manuscript publicized April 5, 2017.

[†]The author is with National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, 135–0064 Japan.
[††]The author is with National Institute of Informatics (NII), Tokyo, 101–8430 Japan.
[†††]The author is with University of British Columbia, British Columbia, Canada.
[††††]The authors are with Toyota Technological Institute (TTI), Nagoya-shi, 468–8511 Japan.
a) E-mail: lihua.zhao@aist.go.jp
b) E-mail: ichise@nii.ac.jp
c) E-mail: zheng.liu@ubc.ca
d) E-mail: smita@toyota-ti.ac.jp
e) E-mail: yutaka.sasaki@toyota-ti.ac.jp
DOI: 10.1587/transinf.2016EDP7337

Sensor data and digital map information should be represented in the format that autonomous vehicles can understand. Therefore, we use ontologies to represent relevant knowledge, which is defined as an explicit specification of a conceptualization [1]. Ontologies are the structural frameworks to organize information, which have been used in Artificial Intelligence, Semantic Web, and Biomedical Informatics as a form of knowledge representation about the world or some part of it. The Semantic Web is an extension of the World Wide Web that enables interoperation between systems by sharing data [2]. *Resource Description Framework* (RDF) is recommended by the *World Wide Web Consortium* (W3C) for describing concepts in data modeling [3]. An ontology mainly consists of concepts (classes) and the relationships (properties) among them. An instance is described by a collection of RDF triples in the form of <subject, property, object>, where property is also called predicate [4]. *Web Ontology Language* (OWL) is also a semantic markup language developed as a vocabulary extension of RDF, which has more vocabularies for describing classes and properties [5].

To represent the stream data from the sensors equipped on the autonomous vehicles, we use timestamp-based temporal RDF representation to construct RDF Stream data [6]. Furthermore, to access the ontology-based data, we use *SPARQL Protocol and RDF Query Language* (SPARQL), which is a powerful RDF query language for accessing to static RDF data [7]. C-SPARQL is an extension of SPARQL designed to express continuous queries such as RDF stream data [8]. 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 [9].

The main purpose of constructing ontologies and ontology-based knowledge base is to enable autonomous vehicles to understand the detected objects from sensor data and to make appropriate driving decisions in different situations. *Advanced Driver Assistance Systems* (ADAS) are designed to improve car safety by perceiving a driving environments and making decisions for safe driving. We constructed ontology-based knowledge base for developing a driving decision making system by adding ontology-based traffic regulations. The driving decision making system can aware dangerous situations at real-time and send warning signals to avoid overspeed or collision. We assume that the received sensor data of environment detection and collision warning detection is accurate.

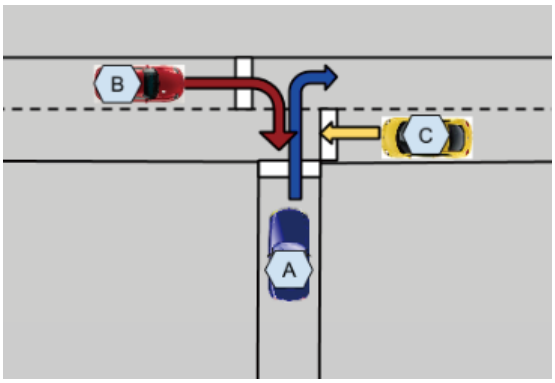


Fig. 1 Uncontrolled intersections and narrow roads.

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. In Fig. 1, car C has the highest priority because it's running straight. Then car A has higher priority than car B, because it is going out from a narrow road to a wider road[†]. Furthermore, the narrow road 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, in this paper, we propose 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.

The remainder of this paper is organized as follows. In Sect. 2, we review some related works, which also utilize ontologies for assisting vehicles. We introduce the ontology-based knowledge base for autonomous vehicles, which contains ontologies, instances of maps and paths, and traffic rules in Sect. 3. In Sect. 4, we describe the decision making system that utilizes the ontology-based knowledge base to improve safety for autonomous vehicles. Section 5 presents and discuss the experimental results with both simulation and real-world data. We conclude this research work and propose future work in Sect. 6.

2. Related Work

Ontologies have been applied in the intelligent transportation field to describe semantic knowledge of the traffic scenarios and to improve safety at intersections. In this section, we review related research on ontologies for intelligent vehicles.

A description logic knowledge was presented for vision-based intersection understanding [10]. They developed a knowledge base for road networks, which contains TBox for describing general knowledge about road networks and ABox for capturing partial information about a particular road or intersection. TBox is a set of terminological axioms and ABox is a set of assertional axioms. The incomplete sensor data was fused with the knowledge base to detect inconsistency in the sensor data.

An ontology-based traffic model was introduced to represent typical traffic scenarios such as intersections, multi-lane roads, opposing traffic, and bi-directional lanes [11]. Relations such as opposing, conflicting, and neighboring were employed to represent the semantic context of the traffic scenarios for decision making. The traffic model was used as a basis for an autonomous vehicle simulator, which was proved to be beneficial for implementing and evaluating different driving behaviors without low-level trajectories. However, the model does not contain semantic information such as speed limitation or actual geometric information of the roads.

A complex intersection ontology was introduced to enable vehicles to aware driving situations, which contains concepts of car, crossing, road connection (lane and road), and sign at crossing (traffic light and traffic sign) [12]. The authors constructed a lean ontology to facilitate fast reasoning about traffic rules for involved vehicles. As a restriction, the passing destination road over the intersection should be predefined for each vehicle. Furthermore, the inference rules were not tested with simulation or real-time test, and the computation time ranges from 1.1s to 3.6s in heavy traffic situations.

Two ontology models about autonomy levels and situation assessment for *Intelligent Transportation Systems* (ITS) were applied for co-driving [13]. One ontology defined the relationship between the automation levels and the algorithmic needs. Another ontology was related to the situation assessment level including the concepts of driver, ego-vehicle, communication, free zone, moving obstacles, and environment. Inference rules were 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. Inference rules were proposed to compute the automation level for each specific automated vehicle. This paper did not present experimental results by embedding the ontologies into real platforms (CyberCars).

An ontology-based context awareness ADAS was developed to enable vehicles to understand the interactions between perceived entities and contextual data [14]. The ontology contains context concepts such as mobile entity, static entity, and context parameters. This machine-understandable ontology enables the vehicle to understand the context information when it approaches road intersections. Experiments showed that the vehicle was able to process human-like reasoning on global road contexts with 14 rules written in SWRL. Although their proposed ontology can be used in real time to retrieve the key entities that a driver should consider, they did not evaluate their approach on intersections with other vehicles.

[†]In Japan, cars run on the left-hand side. However, this can be easily adjusted to the other side driving.

QualiTraj ontology was designed to represent time-based characteristics of trajectories [15]. A profile was used to represent a dynamic characteristics of a trajectory, which has a sequence of segments. Each segment has qualitative value such as "Increase", "Decrease", and "Steady". They also introduced the concept of Key Point, which was used to represent location and timestamp attributes. The authors used QualiTraj ontology to represent raw sensor data into qualitative semantic representation.

An ontology for automated driving was constructed to represent context information of static infrastructure and dynamic environment [16]. In order to differentiate relevant traffic objects on the road from other objects like trees and parked cars, algorithms for information aggregation of dynamic traffic objects and a-priori map information were introduced. Their approach allowed fast information access, that can be applied for autonomous driving.

A sophisticated digital map that can represent the details of road networks such as speed limit, allowed driving direction, lane-level information and connected road segments is essential resource for autonomous vehicles to perceive driving environment. Furthermore, an autonomous vehicle needs a trajectory and concepts of different control information to understand driving motions and to make driving decisions. Therefore, we construct ontologies for autonomous vehicles by considering these essential factors to improve safety in autonomous driving, while the above research works only considered each particular aspect. In contrast to above research, we focus on roads in urban areas of Japan. The advantages of our research are as follows:

- By accessing the ontology-based knowledge base, we can retrieve semantic knowledge about driving environments, such as, speed limit, driving direction, lane information, and connected road segment information at real time.
- Right-Of-Way rules written in SWRL are used for inferring rules to make safe driving decisions when the vehicle receives collision warning signals. The traffic rules are general rules rather than specifically designed for special cases.
- Our decision making system is evaluated with both simulation and field test using real-time sensor data.
- The proposed approach can be easily extended to deal with complicated traffic situations by decomposing them into a combination of simple traffic situations.

3. Knowledge Base

A knowledge base that can represent observed environment and map information is essential data resource for autonomous vehicles. Autonomous vehicles should be able to understand the knowledge and infer driving behavior by processing the knowledge. Therefore, we construct machine-understandable ontologies and ontology-based semantic knowledge base for autonomous vehicles, which are called TTI Core Ontologies.

We provide the description of the knowledge base using *Vocabulary of Interlinked Datasets* (VoID) and some of the statistical information are described in Table 1. VoID is an RDF Schema vocabulary for expressing metadata about RDF datasets [17]. There are 37,566 RDF triples, 1,424 entities, 149 classes, and 75 properties (40 data properties, and 35 object properties) in the data dump. The repository of dataset is available on the website "http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Ontology/TTICore-0.1/". The ontologies can represent knowledge of sophisticated maps, paths and driving control concepts that are necessary for autonomous driving [18]. We mainly focused on collecting concepts of road structures in urban areas of Japan, specifically, around Toyota Technological Institute (TTI) campus, in Japan. We also collected different types of roads and places from Wikipedia and driver's handbooks. The control ontology and car ontology only contain the concepts that are required for our experiments. The ontologies can be easily extended by adding new concepts based on our ontology structure.

The structure of semantic knowledge base is shown in Fig. 2, which is based on three main ontologies: map ontology, control ontology, and car ontology. The map ontology is used to create map instances and to define traffic rules. The control ontology is mainly used to describe path instances and also used to create traffic rules. The car ontology is used to create car instances and traffic rules with other ontologies. The path instances are constructed based on the map instances and control ontology. The knowledge base contains these three ontologies, instances about maps, paths, cars, and traffic rules written in SWRL.

3.1 TTI Core Ontologies

Three ontologies are constructed to describe knowledge of maps, control related concepts, and information of cars. The map, control, and car ontologies are "TTIMapOnto.owl",

Table 1 Statistics of the knowledge base.

Name:	Dataset for Safe Autonomous Driving
Homepage:	http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/
Data Dump:	Ontology/TTICore-0.1/
VoID Description:	void.ttl
RDF Triples:	37,566
Entities:	1,424
Classes	149
Data Properties	40
Object Properties	35

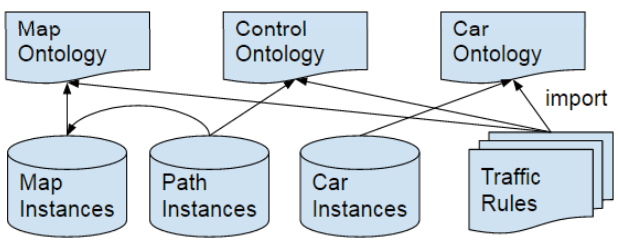


Fig. 2 Structure of the Semantic Knowledge Base.

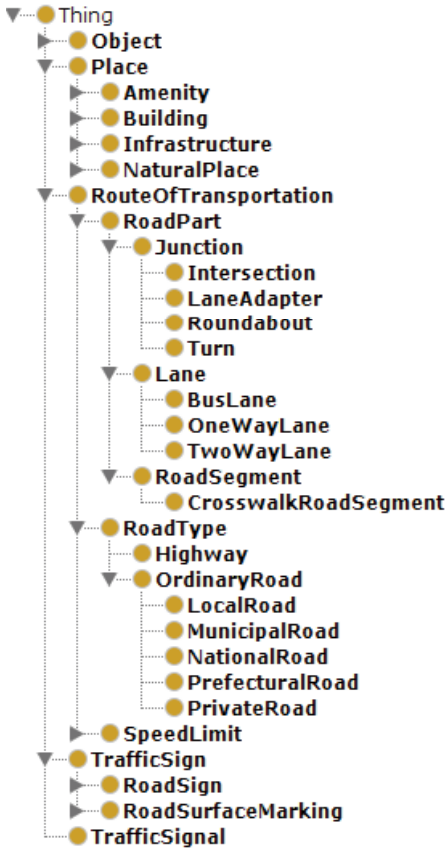


Fig. 3 Main classes of Map ontology.

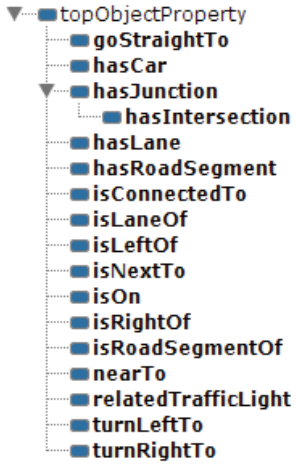


Fig. 4 Object properties of Map ontology.

"TTIControlOnto.owl", and "TTICarOnto.owl", respectively. In the following sections, we describe the main concepts in each ontology, which are currently used for developing ADAS in the Research Center for Smart Vehicles of the Toyota Technological Institute (TTI).

3.1.1 Map Ontology

A sophisticated machine understandable map is required for autonomous vehicles to perceive driving environments. Therefore, we construct a map ontology as illustrated in Fig. 3 and Fig. 4. The map ontology can describe road networks such as road, intersection, road segment, lane, crosswalk, and traffic light information, etc. The map ontology contains 80 classes, 17 object properties, and 33 data properties. We use "*map*:" as the abbreviation of map ontology prefix <http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Map#>.

Figure 3 shows the main classes of map ontology, where the subclasses of "RouteOfTransportation" are mainly used for constructing the road networks. A road consists of junctions and road segments, where a road segment consists of an arbitrary number of lanes. Currently, we have concepts "Intersection", "LaneAdapter", "Roundabout", and "Turn" as the subclasses of "Junction" class. There are three subclasses of "Lane", which are "OneWayLane", "TwoWayLane", and "BusLane". Roads are mainly classified into "Highway" and "OrdinaryRoad", where "OrdinaryRoad" has five subclasses "LocalRoad", "MunicipalRoad", "NationalRoad", "PrefecturalRoad", and "PrivateRoad".

The object properties shown in Fig. 4 are used for describing the relations among the individuals of the classes. The object properties map:goStraightTo, map:turnLeftTo, and map:turnRightTo are used to identify the driving directions when a car drives from one road part to another. We use map:relatedTrafficLight to link the related traffic lights that a driver should observe on a lane. Other properties are used to describe the relations among different road parts.

We also use some data properties to describe speed limit, road attributes, and GPS positions, etc. For example, datatype property map:speedMax is used to describe the speed limit and map:boundPOS is used to describe boundary GPS points of junctions and road segments. We also use map:osm_ref to link the road individuals with OpenStreetMap road entities.

3.1.2 Control Ontology

Control ontology is constructed to describe driving actions and paths of autonomous vehicles. Figure 5 and Fig. 6 show the main classes and object properties included in the control ontology, respectively. We use "*control*:" as the abbreviation of control ontology prefix <http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Control#>. This control ontology contains 35 classes, 15 object properties, and 2 data properties. As shown in Fig. 5, we use instances of control:PathSegment to represent path segments of a trajectory instead of using a collection of GPS points. A path segment can be an intersection, 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 object property control:nextPathSegment to link connected path segments and use the data property control:pathSegmentID to index path segments as shown in Fig. 6. When a vehicle is approaching to an intersection, we

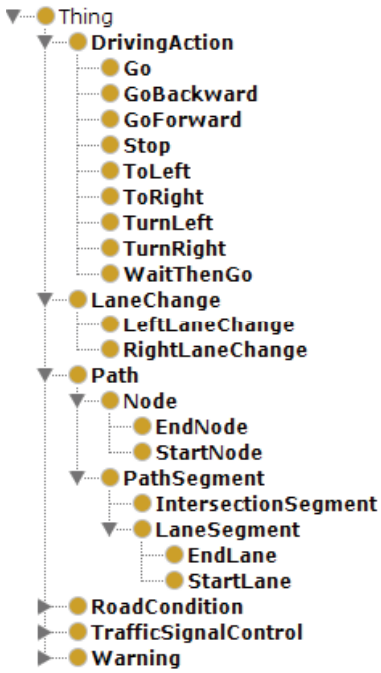


Fig. 5 Main classes of Control ontology.

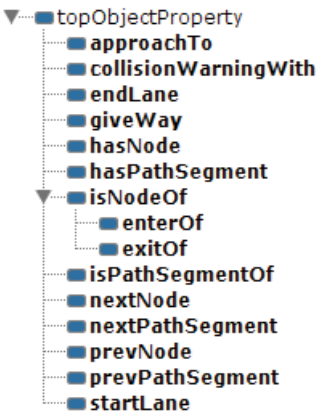


Fig. 6 Object properties of Control ontology.

use control:approachTo to describe the situation. The object property control:collisionWarningWith is defined to indicate upcoming collisions between our vehicle and the other vehicles. We use control:giveWay to represent whether our vehicle should give way to the other vehicles according to the Right-of-Way traffic regulations.

Currently, there are only two data properties: control:nodePos and control:pathSegmentID to represent the position of a node and the index ID of a path segment.

3.1.3 Car Ontology

The car ontology contains concepts of different types of vehicles and the devices, which are installed on a car such as sensors and engines as shown in Fig. 7. This ontology includes 32 classes, 3 object properties, and 16 data properties. We use "*car*:" as the abbreviation of car ontology prefix

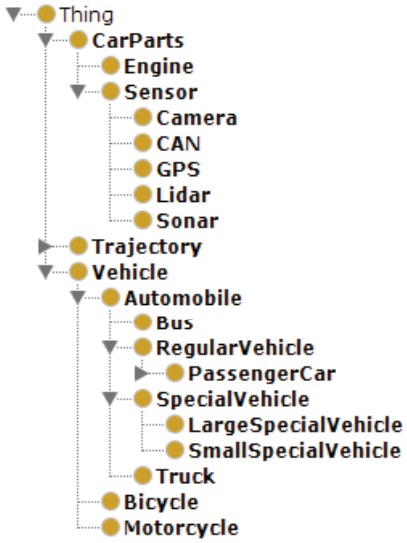


Fig. 7 Main classes of Car ontology.

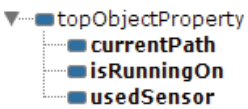


Fig. 8 Object properties of Car ontology.

<http://www.toyota-ti.ac.jp/Lab/Denshi/COIN/Car#>.

The object property car:isRunningOn is used to assert the current location of a car and car:usedSensor is used to describe the sensors installed on the car (Fig. 8).

We constructed many data properties to describe a vehicle, for example, "car_ID", "car_length", and "velocity". The default ID for our car is "0" and we use natural numbers to describe other vehicles' IDs.

3.2 Instances

Instances are also known as individuals that model abstract or concrete objects based on the ontologies. We model instances such as maps, paths, and cars with the ontologies as shown in Fig. 2. We construct map data of a route in Tempaku ward of Nagoya Japan based on the map ontology and control ontology. The statistical information about the map dataset (TTITempakuMapData.owl) is described in Table 2. The map dataset includes 116 intersections, 140 road segments, 5 crosswalks, 316 one-way lanes, 13 two-way lanes, 23 bus lanes, and 330 traffic lights with accurate GPS positions. There is one private road with speed limit of 25km/h, 4 local roads, 2 municipal roads, and 3 prefectural roads.

Figure 9 demonstrates how these road parts are assigned on a map. The road segment (A+B: Hisakata2TTIRoadRS2) is separated into two lanes A (Hisakata2TTIRoadRS2Lane1) and B (Hisakata2TTIRoadRS2Lane2), and it is connected with two intersections H (Hisakata2TTIRoadInt1_2) and C (Hisakata2TTIRoadInt2_3). Figure 10 illustrates the

relations among the individuals of road parts: one local road (Hisakata2TTIRoad), three intersections (Hisakata2TTIRoadInt1_2, Hisakata2TTIRoadInt2_3, Hisakata2TTIRoadInt3_4), two neighbor road segments (Hisakata2TTIRoadRS2, Hisakata2TTIRoadRS3), and four one-way lanes (Hisakata2TTIRoadRS2Lane1, Hisakata2TTIRoadRS2Lane2, Hisakata2TTIRoadRS3Lane1, Hisakata2TTIRoadRS3Lane2). We use the object properties map:hasIntersection and map:hasRoadSegment to link a road with an intersection and a road segment, respectively. We use the object property map:isConnectedTo to link between road segments and intersections, and use map:isLaneOf to relate lanes with road segments as shown in Fig. 10. This figure is expanded by focusing on the intersection C (Hisakata2TTIRoadInt2_3).

Table 3 shows some triples of the instances that are not shown in Fig. 10. The local road Hisakata2TTIRoad relates to the OpenStreetMap entity osm_way:49394393[†], which has speed limit of 40km/h. We use map:boundPos to describe the boundary GPS points of intersections and road segments. For lanes, we use map:enterPos and map:exitPos to indicate the enter and exit positions. In Table 3, the RDF triple <Hisakata2TTIRoadRS3Lane2, control:turnRightTo, TTICampRS1Lane2> indicates that a car which is currently

Table 2 Statistics of the map dataset.

Concepts in Map Ontology	Number of Instances
Intersection	116
Road Segment	140
Crosswalk	5
One-Way Lane	316
Two-Way Lane	13
Bus Lane	23
Traffic Light	330
SpeedLimist25	1
SpeedLimit40	3
SpeedLimit50	6
Prefectural Road	3
Private Road	1
Local Road	4
Municipal Road	2

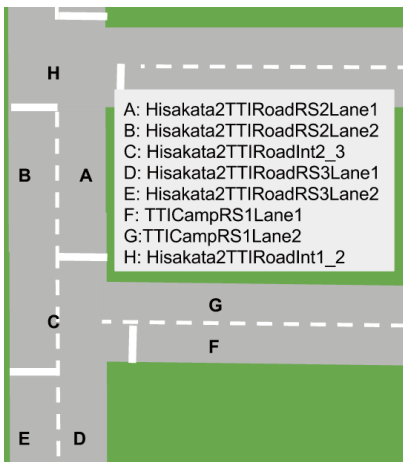


Fig. 9 An example of map structure.

Table 3 Triples of map instances.

Subject	Property	Object
Hisakata2TTIRoad	rdf:type	map:LocalRoad
Hisakata2TTIRoad	map:speedMax	"40"^^kmh
Hisakata2TTIRoad	map:osm_way_id	osm_way:49394393
Hisakata2TTIRoadInt2_3	rdf:type	map:Intersection
Hisakata2TTIRoadInt2_3	map:boundPos	35.107489, 136.982424
Hisakata2TTIRoadInt2_3	map:boundPos	35.107449, 136.982209
Hisakata2TTIRoadInt2_3	map:boundPos	35.107391, 136.982317
Hisakata2TTIRoadInt2_3	map:isConnectedTo	Hisakata2TTIRoadRS3
Hisakata2TTIRoadRS3	rdf:type	map:RoadSegment
Hisakata2TTIRoadRS3	map:boundPos	35.107449, 136.982209
Hisakata2TTIRoadRS3	map:boundPos	35.107329, 136.981323
Hisakata2TTIRoadRS3Lane2	rdf:type	map:OneWayLane
Hisakata2TTIRoadRS3Lane2	map:enterPos	35.107345, 136.981320
Hisakata2TTIRoadRS3Lane2	map:exitPos	35.107462, 136.982203
Hisakata2TTIRoadRS3Lane2	control:turnRightTo	TTICampRS1Lane2
Hisakata2TTIRoadRS3Lane2	control:goStraightTo	Hisakata2TTIRoadRS2Lane2

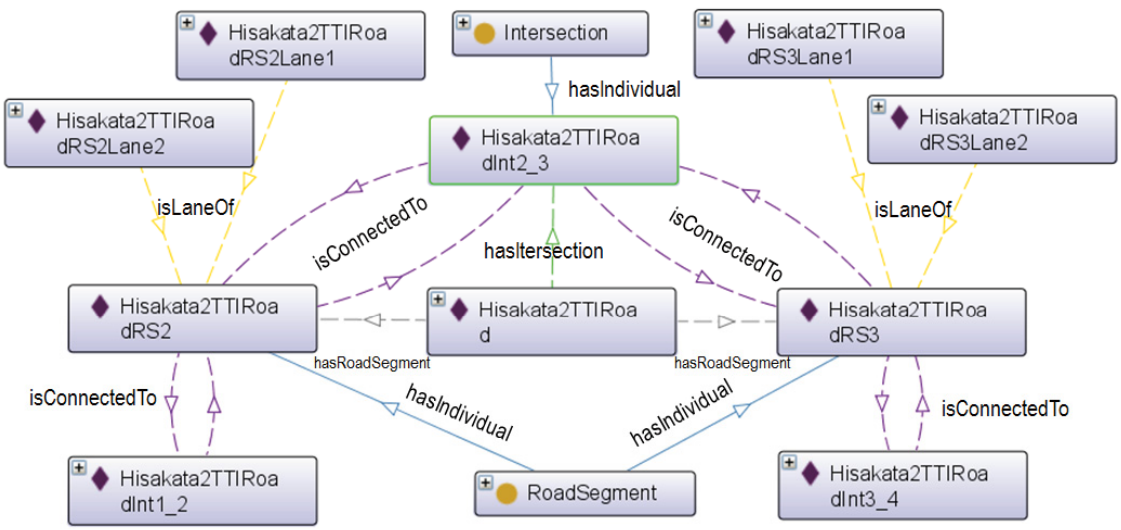


Fig. 10 Visualization of the connection of road parts.

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

Table 4 Triples of the path E-C-G.

Subject	Property	Object
path:ExamplePath	rdf:type	control:Path
path:ExamplePath	control:startLane	Hisakata2TTIRoadRS3Lane2
path:ExamplePath	control:endLane	TTICampRS1Lane2
Hisakata2TTIRoadRS3Lane2	rdf:type	control:StartLane
Hisakata2TTIRoadRS3Lane2	control:nextPathSegment	Hisakata2TTIRoadInt2_3
Hisakata2TTIRoadRS3Lane2	control:pathSegmentID	0
Hisakata2TTIRoadInt2_3	control:nextPathSegment	TTICampRS1Lane2
Hisakata2TTIRoadInt2_3	control:pathSegmentID	1
TTICampRS1Lane2	rdf:type	control:EdnLane
TTICampRS1Lane2	control:pathSegmentID	2

running on E (Hisakata2TTIRoadRS3Lane2) will run on the lane G (TTICampRS1Lane2) if it turns right at the intersection C (Hisakata2TTIRoadInt2_3).

Other than the map data, a vehicle also needs a path instance, which contains connected path segments and their index numbers. In the data dump, we provide two path files: TTIPath (TTIPathData.owl) and YagotoPath (TTIYagotoPathData.owl). The TTIPath contains 101 path segments, which is around TTI campus as described in paper [19]. The system introduced in [19] only checks overspeeding situations without considering traffic rules while running around TTI campus. The YagotoPath contains 140 path segments, which starts from TTI campus and ends at the parking place of an apartment near Yagoto station. These paths are constructed based on the map dataset and each path segment is assigned an integer number starts from zero. An example of path for E-C-G is described in Table 4, which consists of three path segments indexed from 0 to 2.

A ToyotaCar instance file (TTICarData.owl) is also provided as an example, which contains the information of carID, model, and equipped sensors. Additional information such as height, length, or wheel base can be added if necessary.

3.3 Traffic Rules

The perceived knowledge of driving situation and environment should be processed and used for making driving decisions by following traffic rules. At intersections, the vehicles should follow Right-of-Way rules and cross safely by avoiding collisions. We used the *Semantic Web Rule Language* (SWRL) to express Right-of-Way rules. We constructed 14 SWRL rules in the knowledge base to handle Right-of-Way rules at uncontrolled intersections and on narrow two-way roads. The SWRL based Right-of-Way rules were designed based on Japanese driving license guide book. For example, vehicles going straight or turning left should have higher priority than the vehicles turning right when they are approaching the same intersection. These traffic rules can be reused to express more complicated traffic rules, for example, at controlled intersections by adding traffic light rules.

Table 5 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. Rule 2 in Table 5 infers carX's driving direction as control:TurnRight by considering the relations between lanes. We have similar rules for control:GoForward and control:TurnLeft.

Table 5 Examples of SWRL rules.

1	collisionWarningWith(?carX, ?carY) ⇒ CollisionWarning(?carX) ∧ CollisionWarning(?carY)
2	Intersection(?int) ∧ isRunningOn(?carX, ?lane1) ∧ turnRightTo(?lane1, ?lane2) ∧ nextPathSegment(?lane1, ?int) ∧ nextPathSegment(?int, ?lane2) ⇒ TurnRight(?carX)
3	CollisionWarning(?carX) ∧ CollisionWarning(?carY) ∧ GoForward(?carY) ∧ TurnRight(?carX) ⇒ Stop(?carX) ∧ giveWay(?carX, ?carY)
4	MyCar(?car1) ∧ isRunningOn(?car1, ?int) ∧ Intersection(?int) ∧ collisionWarningWith(?car1, ?car2) ⇒ Stop(?car1) ∧ giveWay(?car1, ?car2)
5	TwoWayLane(?lane) ∧ isRunningOn(?carX, ?lane) CollisionWarning(?carX) ∧ CollisionWarning(?carY) ⇒ ToLeft(?carX), giveWay(?carX, ?carY)

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.

4. Ontology-Based Decision Making System

The ontology-based Knowledge base is constructed to enable autonomous vehicles to understand the knowledge of driving environment. Therefore, we developed a decision making system, which accesses to the ontology-based knowledge base to make driving decisions such as "Stop", "ToLeft", or "Give Way", etc.

Figure 11 shows the diagram of the decision making system. The cloud shape represents data (whole knowledge base and a temporal sub-knowledge base(SubKB)). The other rectangle shapes represent functions and the arrow lines represent processing steps. The dotted arrows are processing steps in special cases such as when the vehicle changes from one road part to another or when it receives collision warning signals. The main processing steps of the system include:

1. The sensor data receiver gets sensor data via the sensor data transmitter, which also sends detected vehicle's information while there is a potential collision. The sensor data are converted into RDF stream data format with the ontology.
2. The C-SPARQL engine observes the sensor stream data to detect overspeeding situations.
3. The sensor data receiver sends the data to the SPARQL query engine.

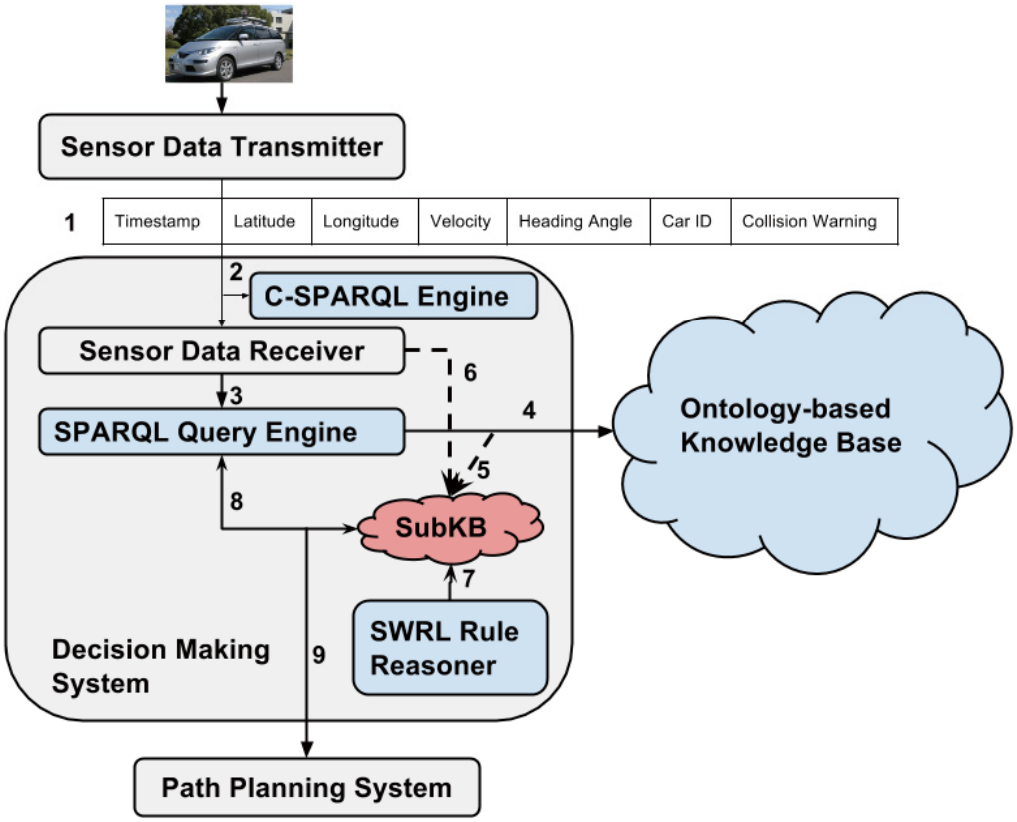


Fig. 11 Diagram of decision making system.

4. The SPARQL query engine accesses to the knowledge base to retrieve information of our vehicle's current lane, next lane, and driving direction, etc.
5. If the vehicle is at the point where it changes from current path segment to the next path segment, we recreate the temporal sub-knowledge base (SubKB), that only contains nearby road segments and the SWRL rules. By narrowing down the searching space of knowledge base, we can significantly reduce rule reasoning time for decision making.
6. When the sensor data receiver gets collision warning signal, we add the driving situation information such as collision warning and the other vehicle's position, velocity, and driving direction into the SubKB. For example, if we detected that our vehicle (carX) has a collision warning with carY, we add a triple <carX, control:collisionWarningWith, carY> to the SubKB.
7. The SWRL rule reasoner performs reasoning on the updated SubKB and new inferred information is added to the SubKB. For example, decisions such as "Stop", "ToLeft", or "Give Way" with the other vehicle's ID.
8. The SPARQL query engine accesses to the inferred SubKB to retrieve the decisions and the vehicles that our vehicle should give way to.
9. The decision signals are sent to the path planning system via the sensor data transmitter to update driving path or driving behavior. Newly added inferred knowledge is removed from the temporal SubKB.

The decision making system mainly consists of a sensor data receiver, a C-SPARQL engine, a SPARQL query engine, a temporal ontology-based SubKB, and a SWRL rule reasoner. In the following, we describe each component in detail.

4.1 C-SPARQL Query Engine

RDF stream data are represented in the format of RDFQuadruple <Subject, Property, Object, Timestamp>. For example, the RDFQuadruple <ex:ToyotaCar, car:velocity, "11.11"^^xsd:float, 1406360324543> represents the velocity of a ToyotaCar at time 1406360324543, where the timestamp uses the time in milliseconds. We use geo:lat[†] and geo:long to represent the latitude and longitude information from the GPS sensor.

We register the OverSpeedCheck C-SPARQL query as shown in Table 6, to query on the RDF stream data. The C-SPARQL query checks if a car's average velocity in the past 500ms exceeds its own allowed maximum speed (maxSpeed, 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.

[†]PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>

Table 6 C-SPARQL query for checking if a car overspeeds its speed limit.

```
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 )
```

4.2 SPARQL Query Engine

The SPARQL query engine contains many predefined SPARQL queries that are used to retrieve knowledge from the knowledge base. SPARQL is a powerful RDF query language that enables Semantic Web users to access to the ontology-based knowledge base. The query that retrieves connected road segments is introduced in the previous section. Here, we will introduce some queries that are used to retrieve road map information and decision result inferred using the SWRL rule reasoner.

The following SPARQL query in Example 1 is used to retrieve the next path segment with two variables - current path segment (tempaku:currPS) and current pathSegmentID ("currentID"^^xsd:int). The first pathSegmentID is 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 when it's revisited.

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

To retrieve the maximum allowed speed of current path segment, we use the SPARQL query in Example 2. The path segment can be either a road segment or an intersection.

```
Example 2:
SELECT ?max
WHERE {{
 tempaku:currPS   map:isLaneOf          ?roadsegment.
 ?roadsegment     map:isRoadSegmentOf   ?road.
 ?road            map:speedMax          ?max. }
UNION {
 ?road            map:hasIntersection   map:currentPathSegment.
 ?road            map:speedMax          ?max. } }
```

The following SPARQL query in Example 3 is used to retrieve all the cars that our experimental car (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 subKB when the decision making system infers according to the Right-of-Way rules in SWRL.

```
Input  : currRS # Current Road Segment
Output: SubKB # Sub-Knowledge Base
dirList ← getConnectedRS(currRS);
rsList ← dirList;
SubKB ← ∅
foreach rs ∈ dirList do
  | rsList.add( getConnectedRS(rs) );
end
foreach rs ∈ rsList do
  | SubKB.add( getAllInfo(rs) )
  | if <rs, map:hasLane, lane> then
  |   | laneList.add( lane )
  | else
  | end
end
foreach lane ∈ laneList do
  | SubKB.add( getAllInfo(lane) )
end
SubKB.add( SWRLRules )
return SubKB
```

Algorithm 1: Sub-Knowledge Base construction.

```
Example 3:
SELECT DISTINCT ?cars
WHERE { car:ToyotaEstima  control:giveWay  ?cars. }
```

We also have a SPARQL query, which retrieves a target node GPS position for updating when the car changes from one road segment to the next one. A target node is defined as the map:exitPos of a lane if the car is currently running on a lane, or map:enterPos of the next lane if the car is currently running on an intersection. The distance of the car and a target node is calculated at real-time according to the Haversine Formula [20]. When the car approaches to the target node, we retrieve the next target node using SPARQL query and update it. We also constructed other SPARQL queries to retrieve the types of a path segment, get triples of an instance, and relations between instances, etc. Jena API[†] is used for executing SPARQL queries on the knowledge base.

4.3 Sub-Knowledge Base (SubKB) Construction

The ontology-based knowledge base contains maps, predefined path, and traffic rules in SWRL. The map will be extended and more traffic rules will be added, which will increase the reasoning time to make a decision. In order to reduce reasoning time, the decision making system should access to a small portion of the knowledge base that is relevant for current driving situation. Therefore, we create a temporal Sub-Knowledge Base (SubKB), which only contains nearby map information.

Algorithm 1 describes the procedure of constructing a temporal *SubKB* based on current position. The current road segment *currRS* can be a lane or an intersection. At first, we get a list of road segments (*dirList*) directly connected to *currRS* using the following SPARQL query in function

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

getConnectedRS(currRS). Then, we get the connected road segments for each road segment *rs* in the *dirList* and put them all in the *rsList*.

```
SELECT DISTINCT ?connectedRS
WHERE {
{ currRS map:isLaneOf ?rs .
  ?rs map:isConnectedTo ?connectedRS. }
UNION
{ currRS map:isConnectedTo ?connectedRS. } }
```

For each road segment *rs* in the *rsList*, which are connected to *currRS* within 2-depth, we get all the triples of them and put in the temporal *SubKB* using *getAllInfo(rs)*. If the road segment has lanes, we put all the lanes into the *laneList*. Then, for each lane in the *laneList*, we get all the triples of the lane instance and put them into the *SubKB*. The function *getAllInfo(rs)* retrieves all the triples of the instance *rs* from the knowledge base. We also add all the SWRL rules into the SubKB and return it as a temporal knowledge base for current position. In Fig. 9, if our vehicle is running on the lane E, the SubKB contains all the information of 2-depth connected lanes (A, B, D, E, G, F, and lanes below E), road segments (A+B, D+E, F+G, and the road segment below D+E), and intersections (C and an intersection below D+E).

4.4 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. The SWRL rule reasoner performs reasoning on the temporally created knowledge base SubKB rather than the whole knowledge base. Since we mainly focus on uncontrolled intersection and narrow road cases, the following situations may occur:

- 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.
- 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 [21]. Pellet API[†] and OWL API[††] are applied for reasoning. According to surveys, Pellet is an open source, which supports SWRL rules and OWL API [22].

[†]http://clarkparsia.com/pellet/
[††]http://owlapi.sourceforge.net/

5. Experiment

In this section, we discuss the experimental results of the ontology-based decision making system. First, we will introduce the experimental area, and the sensor data which are transmitted through User Datagram Protocol (UDP) [23] 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 at all possible situations. Then, we test with the real-time sensor data sent from the sensor data transmitter by driving an Estima car on the predefined driving path. Figure 12 shows the PreScan simulator car and Fig. 13 shows the intelligent Estima car, which was used for our experiments.

5.1 Experimental Area

The experimental path starts from Toyota Technological Institute (TTI) and ends at TTI resident apartment near Yagoto station in Nagoya city of Japan. The following experimental data are from one part of the Yagoto area, which contains both an uncontrolled intersection and a narrow two-way lane as shown in Fig. 14. A lane adapter (G: YagotoIshizakaGrandirLaneAdapter1) connects an uncontrolled intersection (A: YagotoIshizakaInt4_5) and a two-way narrow road (H: YagotoIshizakaGrandirRS1). 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, in

Fig. 12 PreScan Simulator Car.

Fig. 13 Intelligent Car.

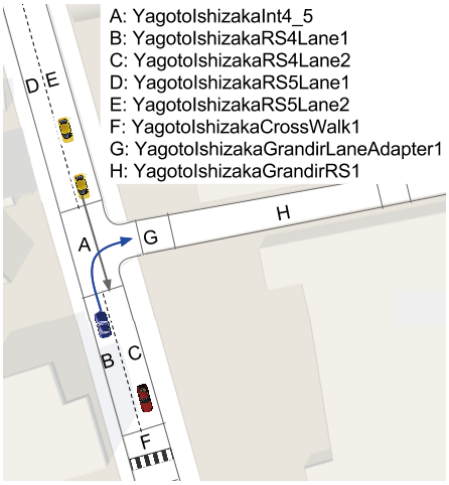


Fig. 14 Experimental area.

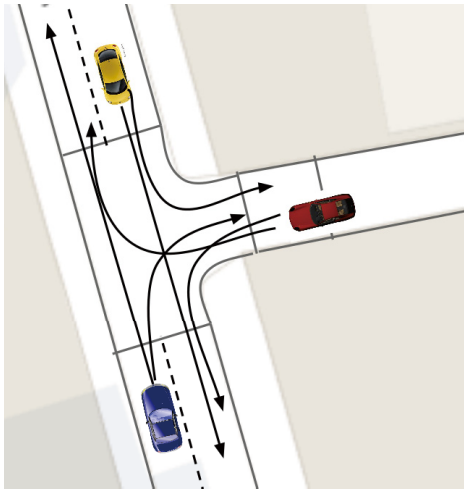


Fig. 15 Uncontrolled intersection

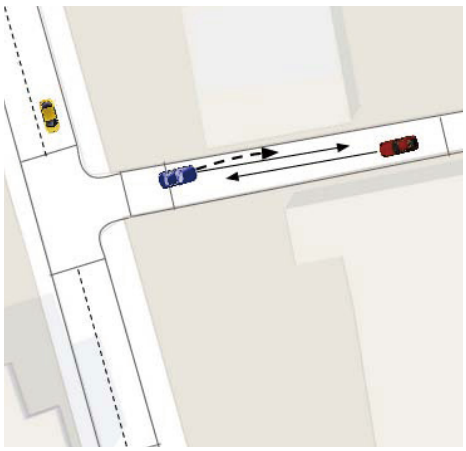


Fig. 16 Narrow two-way lane.

addition to the other vehicles' information, the knowledge of Right-of-Way traffic rules at uncontrolled intersections are necessary for making a decision.

- Even human drivers are cautious when running on two-way narrow roads as shown in Fig. 14. 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.

5.2 Data

Table 7 shows the format of sensor data transmitted at real-time to the sensor receiver of 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.

The instances in the knowledge base for experiments are based on the map ontology and control ontology. The experimental vehicle is assigned a path file for experiments. The instances of other vehicles that have collision warnings with our experimental vehicle are added to the temporal SubKB at real-time and then deleted from the SubKB after driving decisions are inferred.

5.3 Simulation Experiment

To evaluate the ontology-based decision making system in the experimental area as shown in Fig. 14, 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. 15, 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. 16.
- Vehicles: When our vehicle approaches the intersection as shown in Fig. 15, 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 at a different timestamp.

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

Table 7 An example of transmitted sensor data.

Timestamp	Latitude	Longitude	Velocity (m/s)	Heading Angle	Car ID	Collision Warning
1712884	35.13467	136.9641	1.406401	−348.869	0	1
1712884	35.13444	136.9641	3.894356	194.5781	1	1
1712985	35.13467	136.9641	1.478781	−349.403	0	1
1712985	35.1345	136.9639	3.73306	194.0251	1	1
1713076	35.13467	136.9641	1.629769	−350.565	0	1
1713076	35.13473	136.9638	3.611545	194.9413	1	1
1713156	35.13467	136.9641	1.784153	−351.792	0	0
1713237	35.13467	136.9641	1.931167	−353.119	0	0
1713328	35.13467	136.9641	2.074262	−354.594	0	0
1713419	35.13467	136.9641	2.144296	−355.393	0	0
1713510	35.13468	136.9641	2.277657	−357.105	0	1
1713510	35.13482	136.9641	4.012344	194.3415	1	1
1713601	35.13468	136.9641	2.406614	−358.897	0	0
1713783	35.13468	136.9641	2.53862	−0.84693	0	0
1713874	35.13469	136.9641	2.669124	−2.92268	0	0
1713954	35.13469	136.9641	2.726286	−3.95872	0	0
1714045	35.13469	136.9641	2.837927	−6.08027	0	0
1714136	35.13469	136.9641	2.948412	−8.28217	0	0
1714227	35.13469	136.9641	3.055672	−10.6708	0	0

Fig. 17 Experimental results for a path with 3 vehicles.

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 a 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 17 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 (top left) of Fig. 17, 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 the wider 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 other car which is going to drive out from the narrow two-way road. In the last two cases of Fig. 17, 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. 16. This decision is made according to Rule 1 and Rule 5 in Table 5. 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.

5.4 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. A computer is installed in the intelligent vehicle, which contains the ontology-based knowledge base and processes sensor data at real-time. For the real-world data experiment, we hired an experienced driver to drive on the predefined path several times.

The sensor data transmitter sends sensor data in the format as shown in Table 7 while the intelligent vehicle runs on the path as shown in Fig. 18. The path segments for the experiments are B − > A − > G − > H, and so on. Table 8 shows the decisions made in different timestamps according to the data in Table 7. The SWRL reasoner of the decision making system is executed only when the vehicle receives a collision warning. From timestamp 1712884 until 1714045, our vehicle waits and gives way to the other vehicles until the warning is cleared for a specific time period. Here, we assume that the detected vehicles running straight from E(YagotoIshizakaRS5Lane2) to A(YagotoIshizakaInt4_5).

In the following, we describe the decisions and situa-

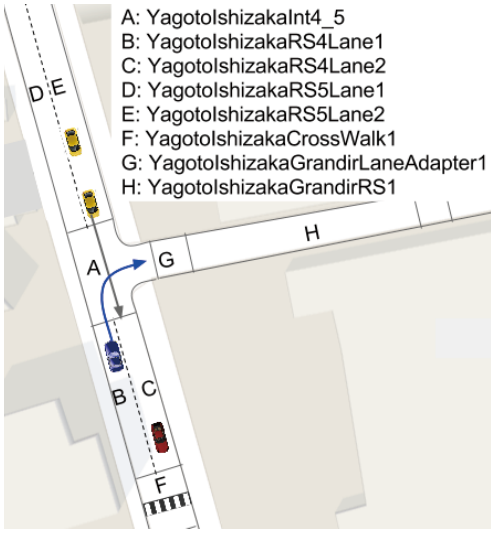


Fig. 18 Before YagotoIshizakaInt4_5.

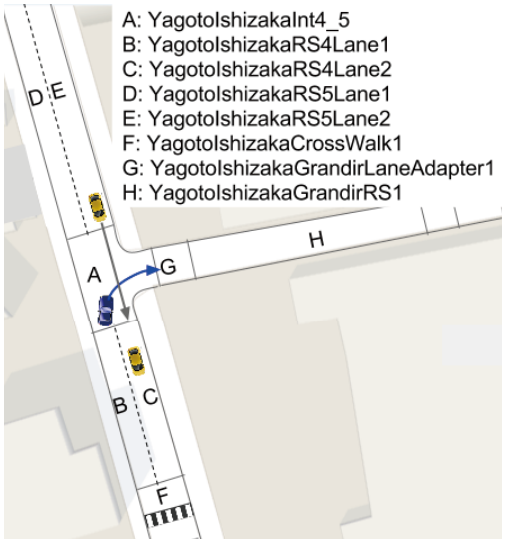


Fig. 19 On YagotoIshizakaInt4_5.

Table 8 Experimental results with real-world data.

Timestamp	Estima Position	Detected Vehicle	Decision
1712884	YagotoIshizakaRS4Lane1	YagotoIshizakaRS5Lane2	Wait, Give Way
1712985	YagotoIshizakaRS4Lane1	YagotoIshizakaRS5Lane2	Wait, Give Way
1713076	YagotoIshizakaRS4Lane1	YagotoIshizakaRS5Lane2	Wait, Give Way
1713156	YagotoIshizakaRS4Lane1	N/A	Receive
1713237	YagotoIshizakaRS4Lane1	N/A	Receive
1713328	YagotoIshizakaInt4_5	N/A	Receive
1713419	YagotoIshizakaInt4_5	N/A	Receive
1713510	YagotoIshizakaInt4_5	YagotoIshizakaRS5Lane2	Wait, Give Way
1713601	YagotoIshizakaInt4_5	N/A	Receive
1713783	YagotoIshizakaInt4_5	N/A	Receive
1713874	YagotoIshizakaInt4_5	N/A	Receive
1713954	YagotoIshizakaInt4_5	N/A	Receive
1714045	YagotoIshizakaInt4_5	N/A	Receive
1714136	YagotoIshizakaInt4_5	N/A	Go
1714227	YagotoIshizakaInt4_5	N/A	Receive

tions in different timestamps.

- **Timestamp:** 1712884 ~ 1713076:
  State: As shown in Fig. 18, the intelligent vehicle is running on B(YagotoIshizakaRS4Lane1) and is going to run on the uncontrolled intersection A(YagotoIshizakaInt4_5). We detected a potential collision with another vehicle, which is running straight from E(YagotoIshizakaRS5Lane2) to A(YagotoIshizakaInt4_5).
  Decision: Wait and give way to the other vehicle.
- **Timestamp:** 1713156 ~ 1713237:
  State: The vehicle is running slowly on the lane B(YagotoIshizakaRS4Lane1) and no collision warning is detected. It keeps waiting for the "GO" decision.
  Decision: Receiving sensor data.
- **Timestamp:** 1713328 ~ 1713419:
  State: The vehicle is running on the intersection A(YagotoIshizakaInt4_5) and no collision warning is detected.
  Decision: Receiving sensor data.
- **Timestamp:** 1713510:
  State: As shown in Fig. 19, the intelligent vehicle is running on the intersection A(YagotoIshizakaInt4_5), and the other vehicle is running straight on E(YagotoIshizakaRS5Lane2).
  Decision: Wait and give way to the other vehicle. (Our vehicle can move if the other vehicle stopped for a specific period, i.e. 500ms, or until receiving five non-collision-warning signals continuously)
- **Timestamp:** 1713601 ~ 1714045:
  State: The vehicle is running on the intersection A(YagotoIshizakaInt4_5) and no collision warning is detected.
  Decision: Receiving sensor data.
- **Timestamp:** 1714136:
  State: We send Go decision if we don't receive collision warning in the following five continuous sensor data.
  Decision: Go.

As the experimental results shown above, the decision making system makes correct decisions at the uncontrolled intersection cases by perceiving driving situations. The vehicle awares the road structure and other vehicle's information that may have upcoming collision with it. By performing reasoning on SubKB, the calculation time is significantly reduced comparing with the system which uses the whole knowledge base [24]. The size of knowledge base used in [24] was about 407kb, while the size of a temporal SubKB in this work is about 19kb ~ 40kb. Therefore, the size of knowledge base for reasoning is reduced to about 1/20 ~ 1/10.

Table 9 shows the calculation time for making a decision using previous system introduced in [24] and our current decision making system with SubKB. The decision making time does not include sub-KB construction time, and the decision making time depends on the performance of ontology reasoner at each situation. Although the construction time for sub-KB depends on the size of the whole KB, it does not affect the decision making time because they

Table 9 Comparison of decision making time.

	Whole Knowledge Base	Sub-Knowledge Base
Maximum	965ms	236ms
Minimum	305ms	37ms
Average	470ms	53ms

are not simultaneous processes. As the comparison result shows, the decision making time depends on the size of the knowledge base. By only considering nearby road segments for decision making, we can significantly reduce the calculation time by providing the same decisions. The average time for making a decision is about 53ms, which is close to the sensor data transmission duration.

6. Conclusion and Future Work

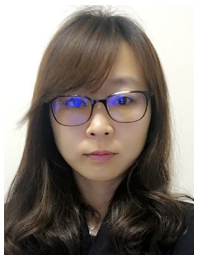
This paper presents that knowledge of driving environment can help autonomous vehicles understand driving situations and make correct driving decisions to avoid collisions. We use machine-understandable ontologies to describe maps and driving situations to enable autonomous vehicles understand the meanings of driving environments. In this paper, we described the ontology-based knowledge base for safe autonomous driving that can be used for developing *Advanced Driver Assistance Systems*. The knowledge base is based on three ontologies: map ontology, control ontology, and car ontology. An ontology-based decision making system is introduced, which can improve safety by making decisions at uncontrolled intersections. Experimental results show that the system can promptly make a decision by using Sub-Knowledge Base and avoid collisions by following Right-Of-Way traffic rules.

We have constructed basic ontologies for autonomous driving and proved that ontology-based knowledge base is applicable for real-time decision making systems. In future work, we will extend the knowledge base and improve the decision making system to deal with more complicated intersections. The knowledge base will be extended to cover more areas of Tempaku ward and include more information such as buildings and subway stations, bus stops, and links to other Linked Open Data sets [7]. The ontologies can be extended by adding concepts of new environments or driving control to deal with various types of driving environments and traffic situations. For example, to make decisions at complicated intersections with traffic lights and varied number of lanes, we can decompose these kinds of complicated cases into several simple situations and integrate the individual results for the final decision. In this paper, we mainly focused on sub-KB construction to reduce the decision making time and did not evaluate the stability of our system. The operational stability can be evaluated by using Field-Programmable Gate Array (FPGA) hardware and we will work on it in future work.

References

[1] T.R. Gruber, "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition, vol.5, no.2, pp.199–220, 1993.
[2] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," IEEE Intell. Syst., vol.21, no.3, pp.96–101, May 2006.
[3] D. Allemang and J.A. Hendler, Semantic Web for the Working Ontologist - Effective Modeling in RDFS and OWL, second ed., Morgan Kaufmann, 2011.
[4] L. Zhao and R. Ichise, "Ontology Integration for Linked Data," Journal on Data Semantics, vol.3, no.4, pp.237–254, 2014.
[5] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein, OWL Web Ontology Language Reference. W3C Recommendation, 2004. http://www.w3.org/TR/owl-ref/.
[6] C. Gutierrez, C.A. Hurtado, and A. Vaisman, "Introducing Time into RDF," IEEE Trans. Knowl. Data Eng., vol.19, no.2, pp.207–218, 2007.
[7] T. Heath and C. Bizer, Linked Data: Evolving the Web into a Global Data Space, Morgan & Claypool, 2011.
[8] D.F. Barbieri, D. Braga, S. Ceri, and M. Grossniklaus, "An Execution Environment for C-SPARQL Queries," 13th International Conference on Extending Database Technology, pp.441–452, ACM, 2010.
[9] SWRL: A Semantic Web Rule Language Combining OWL and RuleML. http://www.w3.org/Submission/SWRL/.
[10] B. Hummel, W. Thiemann, and I. Lulcheva, "Description logic for vision-based intersection understanding," Cognitive Systems with Interactive Sensors, 2007.
[11] R. Regele, "Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles," 4th International Conference on Autonomic and Autonomous Systems, pp.94–99, IEEE Computer Society, 2008.
[12] M. Hülsen, J.M. Zöllner, and C. Weiss, "Traffic Intersection Situation Description Ontology for Advanced Driver Assistance," Intelligent Vehicles Symposium, pp.993–999, IEEE, 2011.
[13] E. Pollard, P. Morignot, and F. Nashashibi, "An Ontology-Based Model to Determine the Automation Level of An Automated Vehicle for Co-Driving," 16th International Conference on Information Fusion, pp.596–603, 2013.
[14] A. Armand, D. Filliat, and J. Ibañez-Guzman, "Ontology-Based Context Awareness for Driving Assistance Systems," Intelligent Vehicles Symposium, pp.227–233, IEEE, 2014.
[15] T.P. Nogueira, R.B. Braga, and H. Martin, "An ontology-based approach to represent trajectory characteristics," 5th International Conference on Computing for Geospatial Research and Application, pp.102–107, IEEE, 2014.
[16] S. Ulbrich, T. Nothdurft, M. Maurer, and P. Hecker, "Graph-based context representation, environment modeling and information aggregation for automated driving," Intelligent Vehicles Symposium, pp.541–547, IEEE, 2014.
[17] K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao, Describing Linked Datasets with the VoID Vocabulary. W3C Recommendation, 2011. http://www.w3.org/TR/void/.
[18] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "Core ontologies for safe autonomous driving," 14th International Semantic Web Conference, 2015. Posters & Demos.
[19] L. Zhao, R. Ichise, S. Mita, and Y. Sasaki, "An Ontology-Based Intelligent Speed Adaptation System for Autonomous Cars," 4th Joint International Semantic Technology Conference, pp.397–413, Springer Berlin Heidelberg, 2014.
[20] C.C. Robusto, "The Cosine-Haversine Formula," The American Mathematical Monthly, vol.64, no.1, pp.38–40, 1957.
[21] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A Practical OWL-DL Reasoner," J. Web Semantics: Science, Services and Agents on the World Wide Web, vol.5, no.2, pp.51–53, 2007.
[22] S. Abburu, "A Survey on Ontology Reasoners and Comparison," International J. Computer Applications, vol.57, no.17, pp.33–39, Nov. 2012.
[23] UDP - User Datagram Protocol. http://ipv6.com/articles/general/User-Datagram-Protocol.htm.

[24] L. Zhao, R. Ichise, T. Yoshikawa, T. Naito, T. Kakinami, and Y. Sasaki, "Ontology-based Decision Making on Uncontrolled Intersections and Narrow Roads," Intelligent Vehicles Symposium, pp.83–88, IEEE, 2015.



Lihua Zhao received her Ph.D. degree in informatics from the Graduate University for Advanced Studies (SOKENDAI), Japan, in 2013. From 2010 to 2013, she worked as a Research Assistant at National Institute of Informatics (NII). After graduation, she worked as a Post-Doctoral researcher at the Research Center for Smart Vehicles of Toyota Technological Institute (TTI) in Nagoya, Japan from 2013 to 2016. Currently, she is a project researcher at the Artificial Intelligence Research Center of National Institute of Advanced Industrial Science and Technology (AIST). Her research interests include semantic web and autonomous vehicles.

Ryutaro Ichise received his Ph.D. degree in computer science from Tokyo Institute of Technology, Tokyo, Japan, in 2000. From 2001 to 2002, he was a visiting scholar at Stanford University. He is currently an associate professor in the Principles of Informatics Research Division at the National Institute of Informatics in Japan. His research interests include semantic web, data mining, and machine learning.

Zheng Liu received his Doctorate degree in engineering from Kyoto University, Japan, in 2000, and the Ph.D. degree from the University of Ottawa, Canada, in 2007. From 2000 to 2001, he was a Research Fellow with Nanyang Technological University, Singapore. He then joined the Institute for Aerospace Research (IAR), National Research Council Canada, Ottawa, ON, Canada, as a Governmental Laboratory Visiting Fellow nominated by NSERC. After being with IAR for 5 years, he transferred to the NRC Institute for Research in Construction, where he was a Research Officer. From 2012 to 2015, he worked as a full Professor with Toyota Technological Institute, Nagoya, Japan. He is now with the School of Engineering at the University of British Columbia (Okanagan). His research interests include image/data fusion, computer vision, pattern recognition, senor/sensor network, condition-based maintenance, and non-destructive inspection and evaluation. He is a senior member of IEEE and a member of SPIE. He is chairing the IEEE IMS technical committee on "industrial inspection" (TC-36). He holds a Professional Engineer license in British Columbia and Ontario. Dr. Liu serves on the editorial board for journals: the IEEE Transactions on Instrumentation and Measurement, Information Fusion, Machine Vision and Applications, and Intelligent Industrial Systems.

Seiichi Mita received his B.S., M.S., and Ph.D. degrees in electrical engineering from Kyoto University in 1969, 1971, and 1989, respectively. He studied at Hitachi Central Research Laboratory, Kokubunji, Japan, from 1971 to 1991, investigating signal processing and coding methods. Currently, he is a professor at Toyota Technological Institute (TTI) in Nagoya (since 1999) and a director of the Research Center for Smart Vehicles at TTI. Currently, he is greatly interested in the research area of autonomous vehicles and sensing systems. Dr. Mita is a member of the Institute of Electronics, Information and Communication Engineers and the Institute of Image Information and Television Engineers in Japan. He is also a member of IEEE. He received the best paper award from the IEEE Consumer Electronics Society in 1986 and the best paper and author awards from the Institute of Television Engineers in Japan in 1987 and 1992, respectively.

Yutaka Sasaki received the B.E., M.Eng., and Ph.D. in Engineering from the University of Tsukuba, Japan, in 1986, 1988, and 2000, respectively. In 1988, he joined the NTT Communications and Information Processing Laboratories, Japan. From 1994 to 2004, he was with the NTT Communication Science Laboratories. From 1995 to 1996, he spent one year at the Intelligent Software Group, Simon Fraser University, Canada as a visiting researcher. From 2004 to 2006, he was with the ATR Spoken Language Translation Research Laboratories. In August 2006, he joined the University of Manchester, UK. From November 2009, he is a professor at Toyota Technological Institute, Nagoya, Japan. His research interests include knowledge-based intelligent systems and statistical natural language processing.