Adjusting Positions of Vehicle Parts based on Rules for Unknown Drivers

Kwangsoo Kim, Bong Wan Kim, Sunwhan Lim, Dong-Hwan Park
IoT Research Division
Electronics and Telecommunications Research Institute
Daejeon, Republic of Korea 34129
Email: {enoch, kimbw, shlim, dhpark}@etri.re.kr

***Abstract*—This paper describes an inference engine used in a system and semantic representation for the system which automatically adjusts the positions of vehicle parts based on rules. The inference engine has rules stored in a knowledge base, which describe the relation between the position of the vehicle part and the driver's body size. The inference engine receives the driver's body sizes in real time, and finds a rule associated with the input values by matching a pattern between them. According to the value defined in the rule, the position of the vehicle part is changed automatically. This rule is automatically modified by learning the relation between the driver's preferred position and body size. The number of selected rules and reasoning time are selected as performance indicators of the inference engine. Also, an ontology is designed to share the development results with others. Automated vehicle parts control system can be used as a method that improves the driver's satisfaction by automatically recommending the driver's preferred position in an environment where many unknown people use the same vehicle like a shared car or a rental car.**

Keywords–Vehicle Part Control; Rule; Inference; Ontology.

I. INTRODUCTION

The future worlds we have seen in science fiction movies are becoming a reality one by one. The Internet of Things (IoT) is making these changes possible. The IoT has launched many smart IoT products and services, making our lives and work easier and more efficient [1]. One of these products is a connected car that has the ability to connect to the Internet. Internet access allows the vehicle to share various data with other devices within the car, as well as devices and services outside the car including other cars or infrastructure. The growing use of connected cars enables a lot of new applications. These applications can be divided into two groups: individual vehicle applications and cooperative applications. Individual vehicle applications are services and contents used in a vehicle to improve the efficiency of vehicle use. They are a navigation to guide a path to a destination, a gas station finder in an area, a vehicle condition or fault diagnosis service, and so on. Cooperative applications are provided by connecting between vehicles or infrastructure. They are a collision warning, intersection movement assistance, a blind spot or lane changing warning, left turn assistance, and so on. Also, those applications can be divided into several categories: navigation, infotainment, safety, diagnostics, and payments. Figure 1 shows the shipment trend of connected cars. The connected car shipment was 6.98 million in 2017 and is estimated to reach 23.87 million by 2022, at a CAGR of 27.9% for the forecasted period [2]. We can see that the connected car shipments will grow continually.

Car sharing is another change occurred in the automotive industry. It is a short-term car rental service positioned between a privately-owned car and public transportation. It intends to enhance traffic efficiency, change existing traffic behavior, improve social problem caused by vehicles, and reduce the negative environmental impacts. By the service, the paradigm for automobiles is shifting from ownership to sharing. The car sharing is quickly increasing in the world. Therefore, it is necessary to develop a connected car service which is suitable for car sharing.

Several automakers provide a memory car seat which allows a driver to save its positions for later recall. As the driver of a car is restricted to the owner or his/her family of the car, it is sufficient that the positions of the car seat are automatically set to previously stored positions, which the driver adjusted before. However, in a car sharing service, it is impossible to automatically recommend the preferred seat position to individual driver because an unspecified driver drives the car. Therefore, there is a need for a method for automatically recommending the position of a vehicle indoor component to unknown drivers.

In this paper, we explain how to automatically adjust the position of vehicle parts (e.g., seat and mirrors) to meet the needs of a driver who uses a car sharing service. An inference engine, rules, and driver's body dimensions are used to recommend the location of the interior parts of a vehicle to the driver, which the driver is satisfied. If the driver does not accept the recommended positions and adjust new positions, then the new positions and the body dimensions are inserted into a learning engine. The learning engine generates new rules, which accept the new positions.

The rest of this paper is structured as follows. Section II describes the system architecture. Section III introduces the performance of the inference engine used in the system. Section IV introduces the semantic representation for sharing the system with others. Finally, Section V concludes this paper.

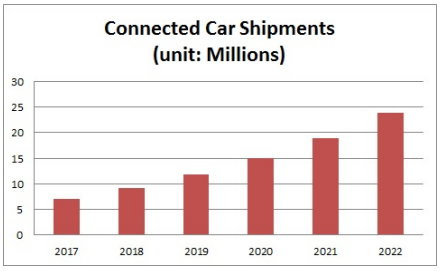

Figure 1. Forecast of connected car shipments (Source: ABI research).

 ISBN: 978-1-61208-658-3

TABLE I. BODY DIMENSIONS AND VEHOCEL PARTS.

Body Dimension	Vehicle Part	Control
height, sitting height, arm length, leg length	driver seat	vertical / horizontal position, tilt
	headrest	vertical position, tilt
	room mirror	left / right position, tilt
	side mirror	left / right position, tilt

II. SYSTEM ARCHITECTURE

A Vehicle Part Position Control System (VPPCS) discussed in this paper consists of an inference engine, rules, a learning engine sensors, and actuators. The inference engine deduces new information by applying logical rules to the knowledge base. The inputs of this engine are rules and body sizes of a driver. The inference engine finds the position of the vehicle part that best suits the driver by pattern matching the driver's body dimensions and rules in real time. The rules describe the relation between body dimensions and positions of vehicle parts. Individual rule stores an initial value that can adjust the position of each vehicle part according to the driver's body size, and the initial value is changed to the optimal value by learning. The learning engine selects and extracts new information from the stored data consisted of body dimensions and positions of vehicle parts. This engine also generates new rules accepting the new information. If a driver adjusts the positions of vehicle parts which are recommended by the inference engine, the learning engine receives the body sizes and the preferred positions of the driver as inputs and executes the learning, and generates new rules. The sensors measure the body sizes (e.g., height) of a driver. The actuators adjust the positions of vehicle parts. The overall system structure is shown in Figure 2, and the body dimensions and position of the vehicle components to be processed in the rule are shown in Table I.

III. INFERENCE ENGINE

In order to apply the VPPCS to a real vehicle, the rule-based reasoner must be able to infer consequences from a set of rules and body sizes in real time. For example, the actuators adjust the position of the drivers seat as soon as the drive is seated. To do this, the reasoner must infer the position of the drivers seat before the driver is seated. In this section, we propose a method to measure the reasoning speed of the inferences engine and to visualize the inferred results.

We use Drools as an inference engine to search for rules controlling vehicle components. Drools is an inference based rule processing engine that uses the Rete algorithm to infer conclusions. The Drools rule consists of a condition and an action, which will be executed if the given condition is true. An example of defined rules is shown in Figure 3.

The advantages of the rule processing engine are as follows [3][4].

- Focus on "what to do" rather than "how to do"
- It is easy to manage and change logics by separating data and logics
- Centralization of knowledge
- It is easy to explain the process which the conclusion is derived

The scenario for applying the VPPCS to an actual vehicle is as follows:

- a driver is close to a vehicle,
- the sensors measure the body size of the driver,
- inferring the positions of the vehicle parts,
- the driver is seated,
- adjusting the position of the vehicle parts,
- the driver adjusts the positions of the vehicle parts if the adjusted positions are inconvenient,
- storing the body sizes and the positions modified by the driver,
- learning using the driver's body sizes and the revised positions,
- producing the new rules reflected the learning results

Therefore, the positions of the vehicle parts deduced by the reasoner must be generated before the driver is seated. In order to measure the reasoning speed of the reasoner, we developed four components. They are a body size emulator that generates a user's body sizes, a set of rules to be used for inference, a rule-based reasoner, and a visualizer to display the reasoning results. In this simulation, 1,122 generated rules are used. The number of conditions included in each rule is ranged from one to four. The target vehicle parts and items of each part are shown in Table I. The range of body sizes used for the initial generation of the rules was the result of the physical examination of the new soldier recruits and the Korean standard body size table published by the Ministry of Commerce, Industry and Energy. The body size emulator randomly generates the height, sitting height, arm length, and leg length. The visualizer displays the number of rules matched with the generated body sizes and rule processing time. As four body sizes are created at one time and each rule contains under four conditions, the number of rules that match the generated

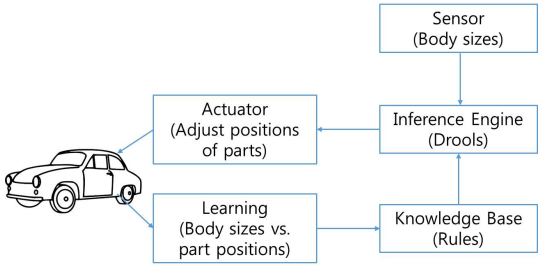

Figure 2. Components of VPPCS.

```
rule "Set Hotisontal Position Under X"
  when
     $v: Value (height < X)      // v: SensingValue Class
  then
     $s.setHorizontalPosition(Y)

  channels["actuator"].send($s)  // s: SeatPosition Class
end
```

Figure 3. Example of position control rules.

values may be four or more. The visualization tool is shown in Figure 4.

The rule-based reasoner is operated in Raspberry PI 2 which is an embedded computer to be used in a real vehicle. In Figure 4, the upper part shows the running time and the lower part shows the session data. The average processing time is 52ms which is shorter than the initial goal, 100ms. The session data includes the identifier of a session, the number of facts, and the number of fired rules.

Figure 5 shows the evaluation environment. We use five types of devices: a camera, a LiDAR sensor, a pressure sensor, actuators, and a Raspberry PI 2. The camera is used to measure a driver's height, the LiDAR sensor measures the distance between a person and the car, the pressure sensor checks whether a driver is seated or not, the actuators control the positions of the seat, and the Raspberry PI is used as a platform which runs the reasoner and the learner. If someone is closer than a given value, the camera takes a picture of the person and extracts his/her height. The height is sent to the reasoner, and the reasoner finds a suitable rule associated with it. The reasoner sends the positions in the found rule to the actuators. Then, the actuators change the vertical position, the horizontal position, and the backrest position of the driver's seat.

IV. Semantic Representation

As this work described in this paper is a part of an open source project, we will release all results developed in the project. The subjects which will be opened are divided into two groups. One is the source code and the other is the system architecture. The source code will be released onto GitHub or a web site and the architecture will be released as an ontology to improve understanding and sharing. Especially, we want to share the control rules with others connected to the Internet. If they exist on the Internet, we want to download them that someone has published. We will revise those to fit our environment and then publish them to the Internet again. If the rules do not exist, we will publish the rules created in our environment to the Internet. Although the automotive ontology was published [5], it does not contain the vehicle part

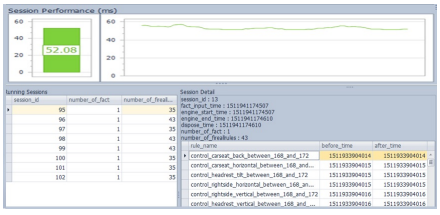

Figure 4. Results of simulation.

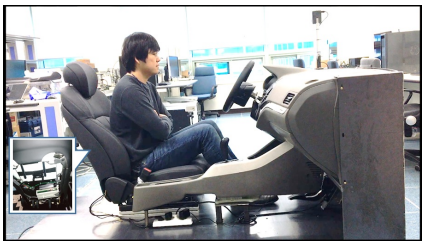

Figure 5. Evaluation environment.

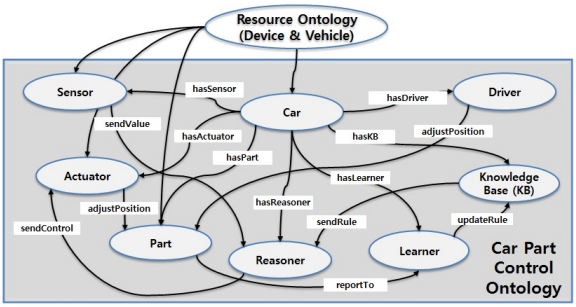

Figure 6. Car part control ontology.

position control. We define a domain ontology related to the vehicle part control. Figure 6 shows the ontology representing the relations among objects. The ontology consists of a car, a sensor, an actuator, a reasoner, a learner, a knowledge base, a vehicle part, and a driver. A driver is inherited from foaf:Person object.

V. Conclusion

In this paper, we describe a vehicle part position control system which will improve the driver's satisfaction in a car sharing service that many unknown people share the same vehicle. An inference engine in the system extracts the positions of vehicle parts from rules according to the driver's body size. Actuators automatically adjusts the positions of the vehicle parts according to the data sent by the inference engine. If the driver adjusts the position, a learner generates a new rule from the driver's body size and the new position. As the performance evaluation index, the number of rules matched the input values and the reasoning time are selected. A total of 1,122 rules are used to measure the inferencing time by using the four body sizes, and the average is recorded as 52 ms. This time is confirmed to be shorter than the initial target value. In the future, further experiments should be conducted to increase the complexity of the rules, to verify the reliability of the system, and to check the scalability when more rules are stored in the knowledge base.

Acknowledgment

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2017-0-00501, Development of Self-learnable common IoT SW engine)

References

[1] J. Choi, J. Park, H. D. Park, and O. Min, "DART: Fast and Efficient Distributed Stream Processing Framework for Internet of Things," ETRI Journal, vol. 39, 2017, pp. 202–212, ISSN: 2-2-3-3-7-3-2-6.

[2] J. Collins and J. Hodgson, Eds., Integrating The Smart Home and The Connected Car. ABI Research, Oct. 2017.

[3] F. Aabedi and G. Etienne, Drools White Paper. LogiCoy, Jan. 2015.

[4] "Business Rules Engines - A White paper," 2012, URL: http://ratakondas.blogspot.kr/2012/06/business-rules-engines-white-paper.html [accessed: 2017-06-02].

[5] M. Feld and C. Muller, "The Automotive Ontology: Managing Knowledge Inside the Vehicle and Sharing it Between Cars," in Proceedings of the 4^{th} International Conference on Automotive User Interfaces and Interactive Vehicular Applications Nov 30–Dec 2, 2011, Salzburg, Austria. Worldwide Publisher, Nov. 2011, pp. 79–86, URL: https://dl.acm.org/citation.cfm?id=2381429 [accessed: 2018-02-15].

 ISBN: 978-1-61208-658-3