An Optimal Driving Support Strategy(ODSS) for Autonomous Vehicles based on an Genetic Algorithm

SuRak Son[1], YiNa Jeong[1] and ByungKwan Lee[1*]
[1] Department of Computer Engineering, College of Engineering, Catholic Kwandong University
Gangneung-si, Gangwon-do, 25601 - Korea
[e-mail: bklee@cku.ac.kr]
*Corresponding author: ByungKwan Lee

Received July 28, 2019; revised August 15, 2019; accepted September 1, 2019; published December 31, 2019

Abstract

A current autonomous vehicle determines its driving strategy by considering only external factors (Pedestrians, road conditions, etc.) without considering the interior condition of the vehicle. To solve the problem, this paper proposes “An Optimal Driving Support Strategy(ODSS) based on an Genetic Algorithm for Autonomous Vehicles” which determines the optimal strategy of an autonomous vehicle by analyzing not only the external factors, but also the internal factors of the vehicle(consumable conditions, RPM levels etc.). The proposed ODSS consists of 4 modules. The first module is a Data Communication Module (DCM) which converts CAN, FlexRay, and HSCAN messages of vehicles into WAVE messages and sends the converted messages to the Cloud and receives the analyzed result from the Cloud using V2X. The second module is a Data Management Module (DMM) that classifies the converted WAVE messages and stores the classified messages in a road state table, a sensor message table, and a vehicle state table. The third module is a Data Analysis Module (DAM) which learns a genetic algorithm using sensor data from vehicles stored in the cloud and determines the optimal driving strategy of an autonomous vehicle. The fourth module is a Data Visualization Module (DVM) which displays the optimal driving strategy and the current driving conditions on a vehicle monitor. This paper compared the DCM with existing vehicle gateways and the DAM with the MLP and RF neural network models to validate the ODSS. In the experiment, the DCM improved a loss rate approximately by 5%, compared with existing vehicle gateways. In addition, because the DAM improved computation time by 40% and 20% separately, compared with the MLP and RF, it determined RPM, speed, steering angle and lane changes faster than them.

Keywords: genetic algorithm, V2C, dig data, WAVE messages, optimal driving strategy

A preliminary version of this paper appeared in The 14th Asia Pacific International Conference on Information Science and Technology (APIC-IST 2019), June 23 - 26, 2019, in Beijing, China. This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) [No. NRF-2018R1A2B6007710].

http://doi.org/10.3837/tiis.2019.12.004 **ISSN : 1976-7277**

1. Introduction

In the auto industry, various researches are conducted to develop self-driving cars that can drive on their own without driver's or passenger's manipulation, to provide drivers with various conveniences. Self-driving cars are classified into six levels by the Society of Automotive Engineers (SAE) International. The 0th stage is non-automated, the 1st stage is driver-assisted, the 2nd stage is partially self-driving, the 3rd stage is conditional self-driving, the 4th stage is advanced self-driving and the 5th stage is complete self-driving [1].

Currently, the conditional self-driving technology, which is in the 3th stage, is becoming more popular and the global companies are developing technologies for advanced self-driving cars, which is in the 4th stage. Self-driving cars are being developed based on various ICT technologies, and the principle of operation can be classified into three levels of recognition, judgment and control. Among them, recognition is the most important technology for self-driving cars. It is important to quickly and accurately recognise the surrounding situation and information so that appropriate judgement and control can be made. The recognition step is to recognize and collect information about surrounding situations by utilizing various sensors in vehicles such as GPS, camera, and radar. The judgment step determines the driving strategy based on the recognized information. Then, this step identifies and analyzes the conditions in which the vehicle is placed, and determines the driving plans appropriate to the driving environment and the objectives. The control step determines the speed, direction, etc. about the driving and the vehicle starts driving on its own. An autonomous driving vehicle performs various actions to arrive at its destination, repeating the steps of recognition, judgment and control on its own [2].

However, as the performance of self-driving cars improves, the number of sensors to recognize data is increasing. The increase of these sensors can cause the in-vehicle overload. Self-driving cars use in-vehicle computers to compute data collected by sensors. As the amount of the computed data increases, it can affect the speed of judgment and control because of the overload. These problems can threaten the stability of the self-driving cars. Thus, to reduce the overload, some studies have developed hardware that can perform deep-running operations inside the vehicle, while others use the cloud to compute the vehicle's sensor data. On the other hand, existing studies use only real-time data such as images and sensor data currently collected from vehicles to determine how the vehicle is driving.

This paper proposes an Optimal Driving Support Strategy (ODSS) which stores historical data in the cloud, reduces the in-vehicle computation by generating big data on vehicle driving within the cloud and determines an optimal driving strategy by taking into account the historical data in the cloud. The proposed ODSS consists of 4 modules. First, a Data Communication Module (DCM) converts CAN, FlexRay, and HSCAN messages of vehicles into WAVE messages and sends the converted messages to the Cloud using V2X communication. Second, a Data Management Module (DMM) classifies the converted WAVE messages and stores the classified messages in a road state table, a sensor message table, and a vehicle state table. Third, The DMM sends the messages stored in these tables to a Data Analysis Module(DAM), which analyzes them to determine the optimal driving strategy by using a Genetic algorithm stored in the Cloud. Fourth, a Data Visualization Module (DVM) displays the optimal driving strategy determined in the DAM and the current driving conditions on the vehicle display for easy recognition by the occupants of the autonomous vehicle.

This paper consists of 5 sections as follows. Section 2 explains the conventional methods related to the ODSS proposed in this paper. Section 3 explains the composition and operation of the ODSS. Section 4 compares the proposed methods with the conventional methods to evaluate the performance. Section 5 explains the conclusion of the proposed paper and future directions of study.

2. Related Work

2.1 Vehicle Machine Learning

Hobold and Silva [3] emphasize Pool boiling the qualification of heat flux using machine learning with an error rate of less than 10%. Visualization windows with the length of single capillary vessel may be enough. Real-time prediction is possible by using a hardware that is small-sized and cheap. Abstract processes with complicated phenomena are common in nature and industry, and many of them are not easy to test mathematically. For instance, Nucleate boiling heat transfer has plenty of applications, but the film boiling is an inappropriate computation technique. Until now, most correlations and computer tests to quantify boiling heat transfer depend on direct measurements of thermal hydraulic data, such as heater temperature, which is often invasive. Here the neural network-based models can quantify heat transfer using only direct and indirect visual information of the boiling phenomenon, without prior knowledge of the governing equations, which enables the non-intrusive measurement of heat flux based on boiling process imaging.

Ning et al. [4] mine the double layers of hidden states of vehicle historical trajectories, and then selects the parameters of Hidden Markov Model(HMM) by the historical data. In addition, it uses a Viterbi algorithm to find the double layers hidden states sequences corresponding to the just driven trajectory. Finally, it proposes a new algorithm for vehicle trajectory prediction based on the hidden Markov model of double layers hidden states, and predicts the nearest neighbor unit of location information of the next k stages.

Li-Jie et al. [5] propose an optional ensemble extreme learning machine modeling technique to improve the wastewater quality predictions, due to the low accuracy and unstable performance of the conventional wastewater quality measurements. An extreme learning machine algorithm is added to the optional ensemble frame as the component model because it runs faster and provides better generalization performance than other machine learning algorithms. The ensemble extreme learning machine model gets over variations in different tests of simulations on a sing0le model. The optional ensemble based on a genetic algorithm is used for ruling out some bad components from all available ensembles to diminish the computation complexity and increase the generalization performance.

Torben et al. [6] propose a supervised machine learning state estimation technique that can evaluate the current side-slip angle of a vehicle. It is composed of a recurrent neural network with gated recurrent units, an supplementary input projection and a regression head. This composition was selected to set a limit on the computational complexity of the model while preserving the expressiveness of the total system. It will also shows how equations of a simplified vehicle model is included to utilize existing domain knowledge. The results show that the neural network can come to an outstanding evaluation while generalizing over various tires, surfaces, and driving situations. Comparisons of different model variants on chosen data

permit us to make a conclusion on the adaptation to varying parameters and show a quality improvement through the physical model.

Yang and John [7] propose a machine learning-based segmentation and classification algorithm, which consists of three phases. The first phase is to preprocess and to prefilter so that it can diminish noise and get rid of clear left and right turning events. The second phase is to employ a spectral time-frequency analysis segmentation technique so that it can generalize all potential time-variant lane-change and lane-keeping candidates. The final phase is to compare two classification techniques, which are dynamic time warping feature with k -nearest neighbor classifier and hidden state sequence prediction with a combined hidden Markov model.

2.2 Vehicle Sensor Data

Feng et al. [8] propose a method to tackle the three-dimensional inverse thermal conduction (3D IHC) problem with the specific geometry of a thin sheet. The 3D thermal equation is simply converted to a 1D equation using modal expansions. By Laplace transformation, algebraic relationships are established which express the front surface temperature and heat flux in terms of those same thermal quantities on the back surface. It expands the transfer functions as infinite products of simplified polynomials using the Hadamard Factorization Theorem. The inverse Laplace transformations of these simplified polynomials draw up relationships for each mode in the time domain. The time domain operations are executed using repeated procedures to compute the front surface quantities from the data on the back surface. The repeated procedures require the numerical differentiation of noisy sensor data accomplished by the Savitzky-Golay method. To deal with the case when part of the back surface can not access sensors, the least squares fit is used to get the modal temperature from the sensor data.

Xiaohui and Junfeng [9] present a two-phase based generous cooperative routing protocol for V2V networks to provide resistance to selfishness. To detect selfish behaving vehicles, a packet forwarding watchdog and an average connection rate based on the multipath weight method are used, where evidence is gathered from different watchdogs. Then, multihop relay decisions are made using a generous cooperative algorithm based on game theory. Finally, through buffering of the multiple end-to-end paths and judicious choice of optimal cooperative routes, route maintenance phase is capable of dealing with congestion and rapidly exchanging traffic.

Ekim et al. [10] propose a new method for consolidating driving behavior and traffic context through signal symbolization. This symbolization framework is proposed as a data reduction method for driving researches. Continuous sensor signals were transformed and reduced into sequences of symbols (chunks) through a hierarchical Dirichlet process hidden Markov model and a nested Pitman–Yor language model. Then, co-occurrence chunking (COOC), the new consolidated method, was applied to the driving behavior and the traffic context chunks. After the consolidation, the COOC chunks were associated with prototype driving scenes by using latent Dirichlet allocation. Finally, the transformed sequence of chunks was clustered into groups.

Gao et al. [11] propose that the cooperative multiple-input-multiple-output (MIMO) and data-aggregation techniques are jointly adopted to reduce the energy consumption per bit in wireless sensor networks by reducing the amount of data for transmission and efficiently using network resources through cooperative communication. For this, a novel energy model is derived that considers the relationship between data generated by nodes and the distance

between them for a cluster-based sensor network using the integrated techniques. Using this model, the influence of the cluster size on the average energy consumption per node can be analyzed.

Liang et al. [12] focuse on how to form clusters with high uniformity and to prolong the network lifetime. For this, it proposes a new clustering scheme named power- and coverage-aware clustering (PCC) to adaptively choose cluster heads according to a hybrid of the nodes' residual energy and loyalty degree. In addition, the PCC technique is not dependent on node distribution or density, which does not have such node hardware limitations as self-locating capability and time synchronization.

Stephanie et al. [13] propose a "no-flow-sensor" wind estimation algorithm for Unmanned Aerial Systems (UAS), which was based on ground speed and flight path azimuth from the autopilot's GPS system. The retrieval accuracy of the predefined profiles by the wind algorithm and its sensitivity to vertical aircraft velocity, diameter of the helical flight pattern and different data sampling methods were investigated. The algorithm with a correspondingly optimized set of parameters was then applied to various scientific flight missions under real wind conditions performed by the UAS Small Unmanned Meteorological Observer(SUMO).

2.3 Vehicle Cloud

Yi-Ke and Li [14] propose a composition-based approach for cloud computing (compositional cloud) using Imperial College Cloud (IC Cloud) as an impirical example. Cloud computing providers/adopters can to design and compose their own systems quickly and flexibly. Cloud computing systems will no longer be fixed, but dynamic and adjustable, depending on the requirements of different application domains.

Bing et al. [15] propose a QoS-aware and quantitative trust-model that is composed of an initial trust value, a direct trust value, and a recommendatory trust value of service, which makes the provision, discovery, and aggregation of cloud services trustworthy. Thus, it can assure trustworthiness of service interoperation between users and services or among services in cloud environment. Simultaneously, service discovery method based on QoS-aware and quantitative trust-model is proposed, which makes a determined trustworthy relationship among a service requestor, a service provider and a service recommender. Users can find reliable service whose overall estimation value is higher.

Sheng and Xu-Cheng [16] propose a synchro-ballistic control approach based on cloud model for the sake of reducing the angle error. First, the mechanism model of steering gear system is introduced. Second, the structure of synchro-control system of twin-rudder is proposed based on the master-slave control strategy. Third, synchro-ballistic controller based on cloud model is designed to solve the nonlinearity and uncertainty of system. Finally, the designed controller is tested via simulation under two different situations.

Yukiko et al. [17] propose a technique for segmenting a 3D point cloud into planar surfaces using recently obtained discrete-geometry results. In discrete geometry, a discrete plane is defined as a set of grid points lying between two parallel planes with a small distance, called thickness. Unlike the continuous case, there appears a finite number of local geometric patterns (LGPs) on discrete planes. Besides, such an LGP does not have the unique normal vector but a set of normal vectors.

Xiaojiang et al. [18] propose to combine the vehicle cloud with the infrastructure-based cloud to expand the currently available resources for task requests on smartphones. In the proposed architecture, the vehicle cloud serves as a cloud service provider for smartphones. In addition, a flexible offloading strategy (FOS) is proposed to perform task migration. The

vehicle cloud can discover and utilize resources that are not utilized in vehicles to accomplish application offloading for smartphones. The FOS estimates the efficiency of various cloud service providers based on current resource conditions and then selects the appropriate cloud service provider to perform the requested task.

3. Design of an Optimal Driving Support Strategy(ODSS) for Autonomous Vehicles based on an Genetic Algorithm

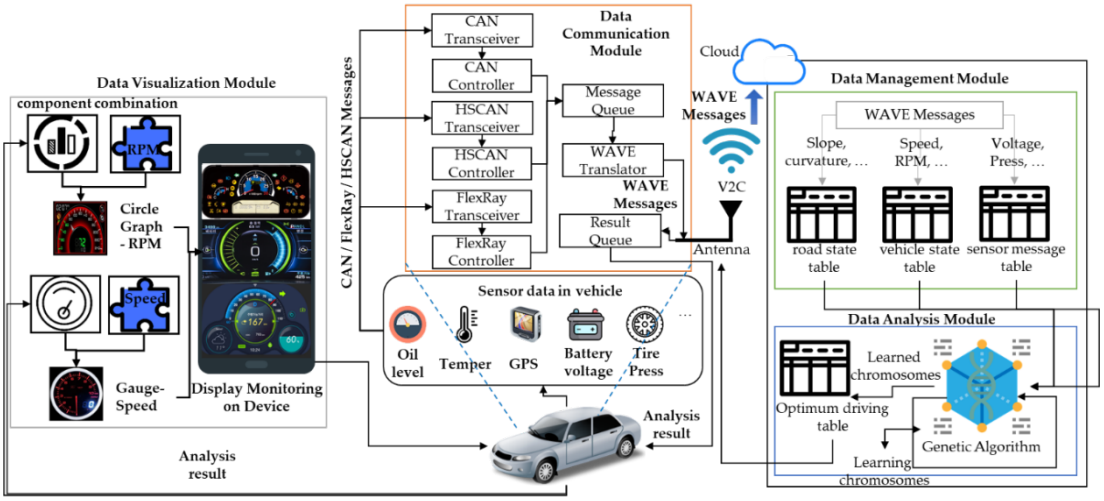


Fig. 1. The configuration of the ODSS

A current autonomous vehicle determines its driving strategy by considering only external factors (Pedestrians, passengers, road conditions, etc.) without considering the interior condition of the vehicle. For example, Adaptive Cruise Control(ACC), which is currently introduced in the second stage of autonomous driving, is an intelligent form of cruise control that slows down and speeds up automatically to keep pace with the car in front of you, but because it does not take into account the vehicle's RPM on a slope, the problem arises that the RPM is not regulated and is maintained at an abnormal speed. To solve the problem, this paper proposes “An Optimal Driving Support Strategy(ODSS) for Autonomous Vehicles based on an Genetic Algorithm” which determines the optimal driving strategy of an autonomous vehicle by analyzing not only the external factors, but also the internal factors of the vehicle(consumable conditions, RPM levels, speed, etc.). The proposed ODSS consists of 4 modules. First, a Data Communication Module (DCM) converts CAN, FlexRay, and HSCAN messages of vehicles into WAVE messages and sends the converted WAVE messages to the Cloud using V2X communication. Second, Data Management Module (DMM) classifies the converted WAVE messages into 3 types and stores the classified messages in a road state table, a sensor message table, and a vehicle state table. Third, The DMM sends the 3 types of messages stored in these tables to a Data Analysis Module(DAM), which analyzes them to determine the optimal driving strategy by using a Genetic algorithm stored in the Cloud. Fourth, a Data Visualization Module (DVM) displays the optimal driving strategy determined in the DAM and the current driving conditions on the vehicle display for easy recognition by the occupants of an autonomous vehicle. **Fig. 1** shows the overall configuration of the ODSS.

3.1 A Design of the DCM

The DCM transmits the measured sensor data to the cloud and the results analyzed in it to the vehicle for visualization on the user's display. **Fig. 2** shows the configuration of the DCM.

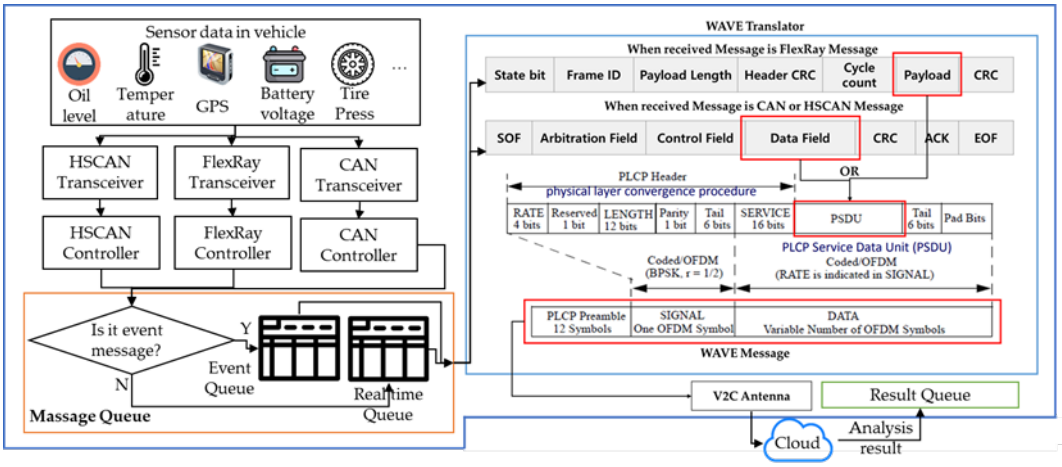


Fig. 2. The Configuration of the Data Communication Module(DCM)

The DCM consists of the Transceiver receiving sensor data, the Controller storing the bus number linked with the sensor, the Message Queue storing the received sensor message, the WAVE Translator that converts the sensor message inside the vehicle into a WAVE message and sends the converted WAVE Message to the Cloud, and the Result Queue that receives the data analyzed in the Cloud and sends the analyzed data to the vehicle [19]. To begin with, the Transceiver receives CAN, FlexRay, and HSCAN messages and sends them to the Controller. The controller stores the bus number of the received messages and sends the messages to the Message Queue. The Message Queue consists of Real-time Priority Multiple Queues and Event Queues and the data transmitted from the sensors is composed of real-time sensing data and event sensing data. Because the event sensing data has to be processed faster than real-time sensing data, the Message Queue processes the message from Event Queue faster, and the messages of the Real-time Priority Multiple Queues are processed according to priority [20].

The Real-time Priority Multiple Queues store priorities separately between protocols and between messages inside the protocol. Because Priorities between protocols are defined in advance, there are no problems caused by priority conflicts. And because the priority of the message is increased according to how long the message stays in Message Queue, the problem that low-priority messages is not processed is solved.

Because the Event Queues process sensing messages for event requests, they have higher priority than the Real-time Priority Multiple Queues. Message Queue sends messages to the WAVE Translator in sequence. The WAVE Translator receives a sensor message from the Message Queue and converts it into a WAVE message. **Fig. 2** shows how the WAVE Translator converts CAN, FelxRay, and HSCAN messages into WAVE messages.

1. The WAVE Translator removes the headers and trailers from the CAN and FlexRay and HS-CAN messages which are received from the Message Queue.

2. After removing the headers and trailers, the WAVE Translator generates WAVE messages by adding headers and trailers to PSDU(Data Field if a received message is a CAN message and HSCAN message, or Payload if it is a FlexRay message).
3. It sends the generated WAVE message through V2C antenna.

When the WAVE messages are accumulated in the Cloud, the Cloud analyzes the driving habits of the vehicle and transmits the analyzed optimal driving strategy to the DCM. The DCM sends the optimal driving strategy analyzed in the Cloud to the Result Queue, which controls the vehicle in a FIFO sequence.

3.2 A Design of the DMM

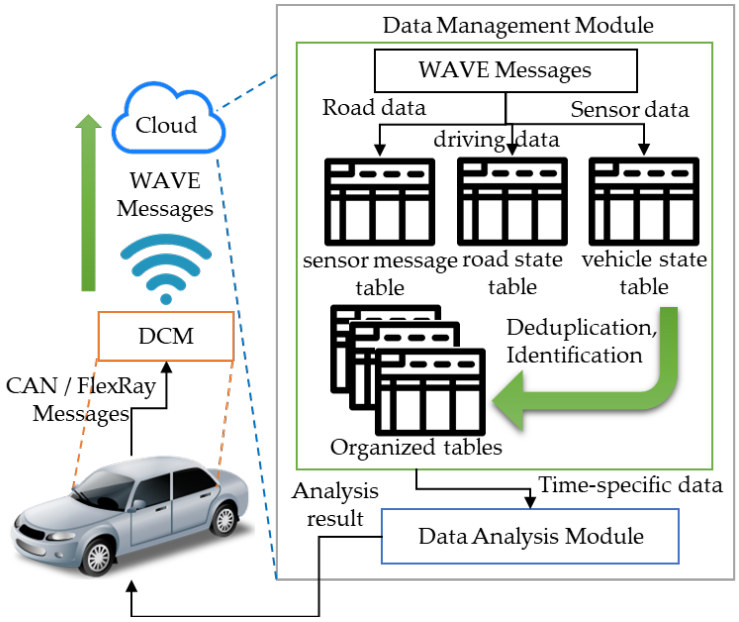


Fig. 3. The configuration of the DMM

The DMM manages WAVE messages sent form the DCM as follows. The DMM generates and manages a road state table that stores the conditions of the road on which a vehicle is currently driving, a sensor message table that stores the vehicle's sensor data, and a vehicle state table that stores the driving state of the vehicle. **Fig. 3** shows the configuration of the DMM and **Fig. 4** shows each information of the 3 tables.

The road state table has the vehicle number, V-Num, the slope of the road on which the vehicle is driving, Slope, the curvature of the road, Curve, and the ID. The ID is given in 50 ms. The DMM stores the vehicle's driving information in the road state table only when the vehicle is driving safely and the data stored in this road state table is used to be learned later in the DAM's genetic algorithm. The WAVE messages received from the vehicle are stored in the sensor message table. The sensor message table has the name of the sensor, Name, the sensor value measured in a vehicle, Data, and vehicle number, V-Num. Finally, the vehicle state table has the speed of a vehicle, Speed, angle of steering, Angle, RPM, and LaneChange. The LaneChange indicates whether the lane is changed or not and the ID is given in 50 ms. The ID value of the vehicle state table has to be the same as that of the sensor message table and the road state table. For example, because the Time value in the Vehicle State table is 2019-05-12 13:24:10.200, the ID is 1. Here, because the Time value in the sensor message

table is 2019-05-12:10.200~10.250 and belongs to the scope of 50ms, the ID is 1. In the same way ID in the road state table is 1.

road state table

Field name	ID	Time	V-Num	Slope	Curve
Data Values	1	2019-05-12 13:24:10.200	66-4371	10.8	15.2
	2	2019-05-12 13:24:10.250	66-4371	10.3	15.1
	3	2019-05-12 13:24:10.300	66-4371	10.1	14.9
	4	2019-05-12 13:24:10.350	66-4371	9.9	14.9

sensor message table

Field name	ID	Time	Name	Data	V-Num
Data Values	1	2019-05-12 13:24:10.231	Oil press	19.1	66-4371
	1	2019-05-12 13:24:10.243	Battery Volt	12	66-4371
	1	2019-05-12 13:24:10.246	Engine Temp	65.1	66-4371
	2	2019-05-12 13:24:10.254	Tire press	32	66-4371

vehicle state table

Field name	ID	Time	V-Num	Speed	Angle	RPM	Lane Change
Data Values	1	2019-05-12 13:24:10.200	66-4371	79	15	2898	0
	2	2019-05-12 13:24:10.250	66-4371	81	15	3068	0
	3	2019-05-12 13:24:10.300	66-4371	76	15	2932	0
	4	2019-05-12 13:24:10.350	66-4371	75	15	2712	0

Fig. 4. Road state table, sensor message table, vehicle state table of the DMM

Because the sensors inside the vehicle differ in both the data measurement cycle and the data measurement time, the DMM should check the duplicates of messages stored in the sensor message table. The DMM classifies the WAVE messages in 50 ms according to the time they are received, and deletes any duplicate WAVE messages in 50 ms. **Fig. 5** shows how the DMM classifies messages in the sensor message table in 50ms and deletes duplicate messages.

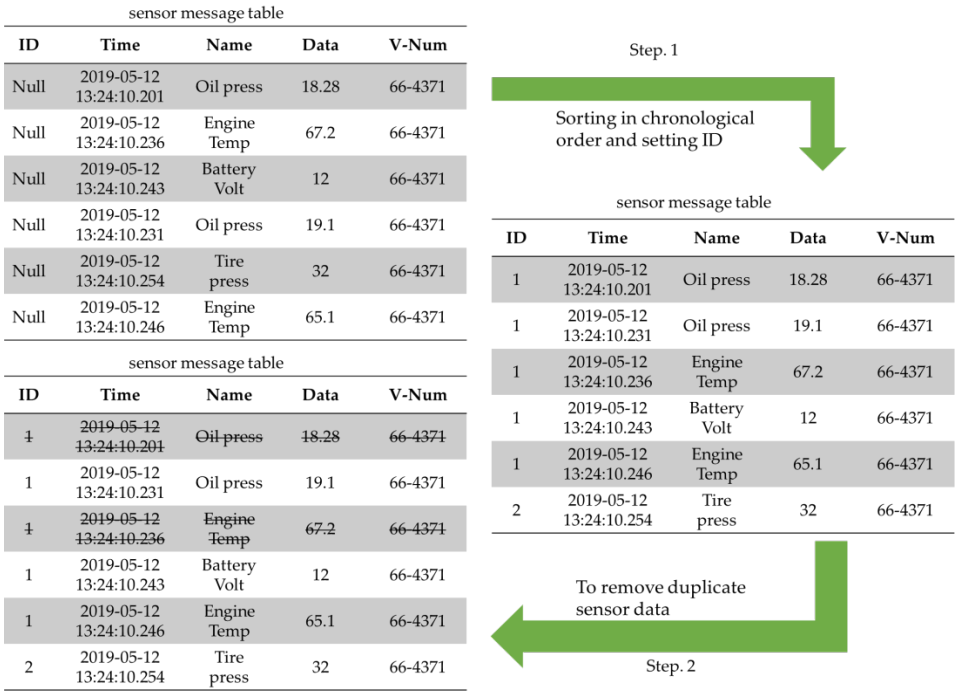


Fig. 5. The process to provide IDs and to remove duplicates in sensor messages table

First, the DMM stores the received WAVE messages in the sensor message table in the order in which they are received. When they are completely stored, it arranges the tuples in chronological order and assigns an ID in 50 ms. In **Fig. 5**, the sensor message table is sorted in chronological order after step1 and the ID of the tuples between 2019-05-12 13:24:10.200 and 10.250 second is set to 1. Next, in step 2, the DMM removes the duplicated Name of among the tuples with ID=1 in the sensor message table and delivers all the tuples of the road state table, sensor message table, and Vehicle state table to the DAM. The DAM executes the genetic algorithm using the tuples it receives from the DMM.

3.3 A Design of the DAM

The DAM analyzes the driving habits of an autonomous vehicle by using a genetic algorithm based on big data stored in the Cloud. The Genetic algorithm generates a gene sequence by using the autonomous vehicle's driving information received from the DMM and determines the optimal driving strategy by using a combination between genes. Because the genetic algorithm requires a lot of data for learning, the safety of an autonomous driving vehicle cannot be guaranteed if a gene sequence is learned only with real-time driving data. To address this, the DAM in this paper can select an optimal driving strategy for autonomous driving because it learns the genes by using the vehicle information accumulated past in the DMM. The DAM consists of a Data Normalization Sub-module (DNS) that normalizes data received from the DMM and a Data Learning Sub-module (DLS) that learns a genetic algorithm.

3.3.1 Data Normalization Sub-module(DNS)

The type of data used by the DAM consists of the training data sets for training a genetic algorithm and the output data analyzed after training. Here, one training data set has a training input and a training output. The training input means the values of the road state table and the sensor message table received from the DMM, and the training output means the values of the vehicle state table. The DNS normalizes data to learn the genetic algorithm. The training data sets used to learn the algorithm are expressed as Formula 1.

$$\mathrm{T} = \left\{\{\mathrm{TX}_1, \mathrm{TY}_1\}, \{\mathrm{TX}_2, \mathrm{TY}_2\}, \{\mathrm{TX}_3, \mathrm{TY}_3\}, \ldots \{\mathrm{TX}_{10000}, \mathrm{TY}_{10000}\}\right\}, \quad (1)$$

Here, TX means a training input and TY means a training output. To learn the genetic algorithm, 10000 of the past data are used and the values for all slopes and curvatures must be the same.

The DNS generates an initial set of chromosomes for RPM, speed, steering angle, and lane change to analyze the training input data. The RPM of the initial set of chromosomes will be selected as 1,000-6000, the speed, 0-250, and the steering angle, 0-90 and whether or not the lane is changed will be selected as 0 or1 randomly. The set of chromosomes generated by the genetic algorithm is evaluated according to the slope and curvature of the current road, and the optimal driving strategy is determined by the slope and curvature of the road as the generation passes. The DAM learns a set of chromosomes for 100 generations and applies a set of learned chromosomes to an autonomous vehicle. Formula 2 represents a set of chromosomes in the genetic algorithm.

$$Y = \{y_r, y_a, y_s, y_{lc}\}, \tag{2}$$

Here, y_r means RPM, y_a means steering angle, y_s means speed, and y_{lc} means whether or not the lane is changed. The DNS normalizes the values of y_r, y_a, y_s and y_{lc}between 0 and 1. When all the values of Y are normalized, y_r, y_a, y_s and y_{lc} have a similar effect on the suitability when assessing the suitability of a set of chromosomes. The DNS normalizes them by dividing RPM by 10000, steering angle by 100 and speed by 1000. Formula 3 shows the process by which each value is normalized. Finally, all values of Y exist between 0 and 1.

$$y_R = \frac{RPM}{10000}, y_A = \frac{angle}{100}, y_S = \frac{speed}{1000}, y_{LC} = \text{LaneChange}, \tag{3}$$

The DNS generates a set of initial 100 chromosomes normalized between 0 and 1 and sends trailing data sets T and a set of 100 chromosomes to DLS.

3.3.2 Data Learning sub-module(DLS)

The DLS learns the genetic algorithm by receiving training data sets and an initial set of chromosomes from the DNS. The DLS uses the genetic algorithm, Algorithm1 to determine the optimal driving strategy of an autonomous vehicle.

Algorithm 1. Genetic Algorithm for the optimal driving strategy of an autonomous vehicle

```
input: radius, slope, bigdata
init : parent[4][4], gene[100][60][4], fitness[10000][60], t=0, roulette;
FOR (int i=0; i<60; i++){
   gene[t][i][0] = random(bigdata.speed);
   gene[t][i][1] = random(bigdata.steeringAngle);
   gene[t][i][2] = random(bigdata.RPM);
   gene[t][i][3] = random(bigdata.laneChange);
}
WHILE(t<100) {
   FOR (int i=0; i<60; i++) {
      fitness[t][i] = f(radius, slope, gene[i]);
      fit = fit + fitness[t][i];
   }

   FOR (int i=0; i<60; i++) {
      roulette[i] = fitness[t][i]/fit;
   }
   FOR (int i=0; i<4; i++) {
     parent[i] = gene[t][random(roulette)]
     gene[t+1][i] = parent[i];
   }
   t ++;
```

```
    FOR (int i=4; i<10000; i++) {
      gene[t][i] = recombine(parent[0], parent[1], parent[2], parent[3])
    }
    mutate(gene[t][i],0.05);
}
Output : gene[t]
```

First, the DLS receives 100 sets of initial chromosomes from the DNS that are randomly set for learning RPM, speed, steering angle, and lane change. The definition of an initial set of chromosomes has already been described in 3.3.1. The DLS computes the suitability by using Formula 4 for 100 sets of randomly generated chromosomes. A typical genetic algorithm compares one training data with a set of chromosomes to find the least different values, or to find the best values among chromosomes. However, various driving strategies on road driving can exist in the same environment. Therefore, the DLS of this paper compares 10000 training outputs with 100 sets of chromosomes to calculate the suitability of a set of chromosomes.

$$\mathrm{Ch}(\mathrm{Y}_k) = -1 * \left| \frac{\sum_{i=1}^{10000} (TY_i - Y_k)}{100} \right|, \tag{4}$$

Here, k is the value of 1 to 100 and $\mathrm{Ch}(\mathrm{Y}_k)$ means the suitability of Y_k. Once the suitability of each chromosome is computed, the DLS sets the probability of selection as much as the suitability of each chromosome, and randomly selects two from 100 chromosomes to combine the two genes. The DLS uses an Arithmetic Crossover method to combine these two genes. The Arithmetic Crossover method computes the average of two sets of parent chromosomes to produce child chromosomes. Formula 5 determines the selected probability according to the suitability of each set of chromosomes, and Formula 6 randomly selects the set of genes by applying $\mathrm{P}(\mathrm{Y_k})$ from the entire chromosome set Y. Formula 7 shows the generation of a child chromosomes.

$$\mathrm{P}(\mathrm{Y_k}) = \frac{\mathrm{Ch}(\mathrm{Y_k})}{\sum_{i=1}^{10000} \mathrm{Ch}(\mathrm{Y_i}) * 100} \tag{5}$$

$$\mathrm{parents} = \mathrm{for}(\mathrm{k} = 1, \mathrm{k} < 100, \mathrm{k}++)\ \mathrm{random}(\mathrm{Y}, \mathrm{Y_k}, \mathrm{P}(\mathrm{Y_k})) \tag{6}$$

$$\mathrm{child} = \frac{\mathrm{parent1} + \mathrm{parents2}}{2} \tag{7}$$

Here, $\mathrm{P}(\mathrm{Y_k})$ means the weight of a set of chromosomes. When a set of 100 child chromosomes is generated by repeating 100 times the process of Formula 5 to 7, the DLS produces a sudden mutation chromosome so that the genetic algorithm outputs a global minimum without falling into the local minimum. The global minimum means the optimal output value that the genetic algorithm can have. The local minimum means the output value that the genetic algorithm has is better than the output value around it, but it does not mean the global minimum. **Fig. 6** shows local mimum point and **Fig. 7** shows mutagenic computation.

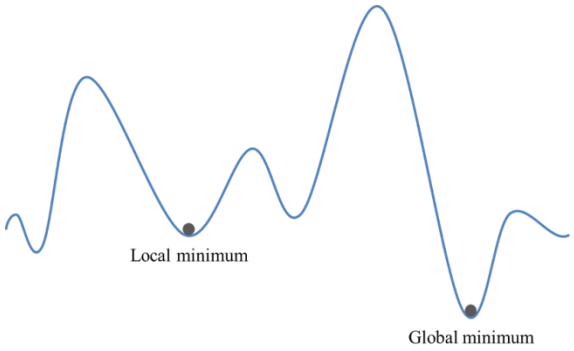


Fig. 6. Local Minimum and Global Minimum Point

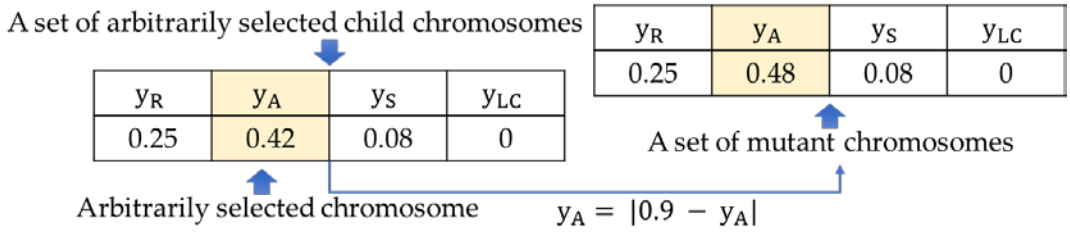


Fig. 7. The generation process of a set of mutant chromosomes

The DLS generates 0.05% of the child genes as mutant genes randomly, and a set of mutant chromosomes by using reverse operations. The DLS randomly selects one of the child gene sets generated in **Fig. 7**, and randomly selects one of the genes from that set. Next, it reverses(in **Fig. 7**, 0.9) the selected gene value (in **Fig. 7**, 0.42) using a fixed maximum value that the gene can have. In this way, the DLS makes 0.05 percent of the total set of genes mutant genes.

When a set of 100 child chromosomes is generated and the mutation operation is finished, the DAM again repeats the mutation computation process from Formula 4. If the learning times are specified up to 1000, and a set of chromosomes with a suitability less than 0.001 is generated, the DLS terminates learning of the genetic algorithm and stores the speed, RPM, lane change, and steering angle for slope and curvature of the road.

3.3.3 Usage of the DAM

The DAM computes an appropriate set of chromosomes for a particular slope and curvature of the driving road, generates an optimal driving table, and stores the set of chromosomes in the optimal driving table. Because the genetic algorithm computes the optimal driving strategy according to specific slopes and curvature by analyzing sufficient historical data, it does not need to perform deep-running, machine-running, etc. operations in real-time. **Fig. 8** shows the optimal driving table and how it communicates with an autonomous vehicle.

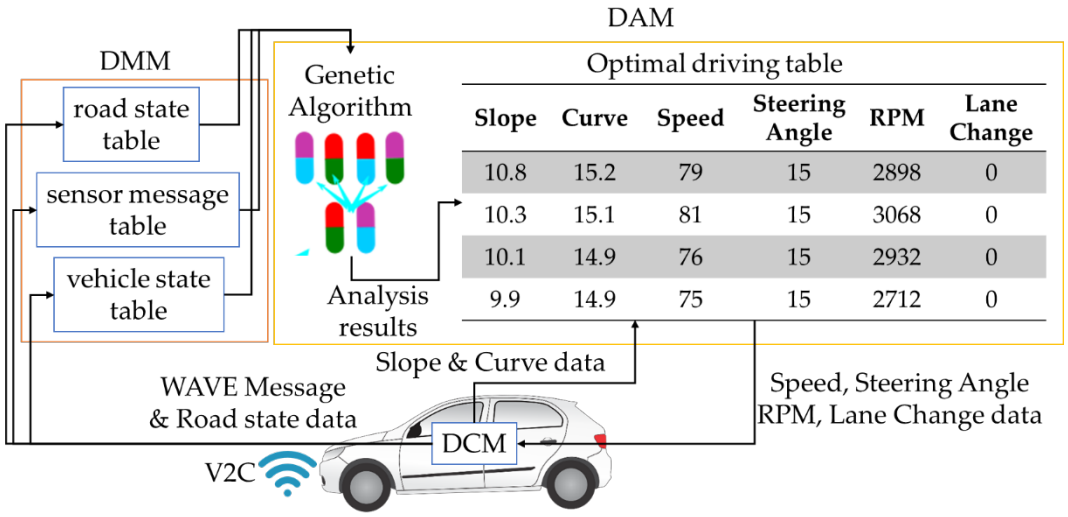


Fig. 8. The way that the Optimal driving table communicate with an autonomous vehicle

The DCM inside the vehicle transmits WAVE messages and Road state data to the DMM using V2C communication. The DMM classifies the received messages and data into 3 types and stores them in the road state table, sensor message table, and Vehicle state table. The DMM transmits the information of 3 tables to the DAM, and the DAM determines the optimal driving strategy by using genetic algorithms. The DAM stores the slope and curvature of the road used in the traing data set in the Slope and Curve field of the optimal driving table and the speed, steering angle, RPM, and lane change of the optimal driving strategy determined using the training data set in the Speed, Angle, RPM, and LaneChange field of the optimal driving table.

A vehicle transmits to the DAM the slope and curve data of the road on which it is driving to the DAM in real time. The DAM searches for a set of chromosomes with similar slope and curvature values in the optimal driving table after inputting the slope and curvature of the vehicle which is currently in driving. If it exists, the DAM transmits it to the vehicle to implement the optimal driving strategy.

3.4 A Design of the DVM

Once the DAM determines the optimal driving strategy of the vehicle, a Data Visualization Module(DVM) visualizes the determined optimal driving strategy and the list of vehicle consumables' condition. The DVM visualises the driving strategy by using road features and the consumable condition by using the shape of a gauge. Therefore, the DVM allows the driver to accurately determine the optimal driving strategy of an autonomous vehicle and when the consumables will be replaced. **Fig. 9** shows the process of DVM receiving data, visualizing it, and placing visualized data on a monitoring device.

The Road Graphics Generator and the Gauge Graph Generator of the DVM visualize the driving strategy of an autonomous vehicle, while the Consumer Graphics Generator visualizes the consumables status of an autonomous vehicle. The Road Graphics Generator uses an arrow to indicate whether the vehicle is going straight or turning to the left or the right depending on the steering angle of the vehicle, and visualizes the lane of the road in which the vehicle is driving and the position of the current vehicle. The Gauge Graph Generator generates Gauge graphs by receiving the vehicle's speed and RPM and places them on the left side of the road

Graphics Generated by the Road Graphics Generator. The Consumables Graphics Generator indicates whether the current state of in-vehicle consumables needs to be replaced or not by receiving sensor data from the vehicle. The Consumerables Graphics Generator measures 10 consumables status, and such specific consummables as Ignition Plug, Air Conditioning Filter that cannot be measured through the sensor measures the exchange period using the distance travelled. The Consumables Graphics Generator displays the consumables that must be replaced immediately in red, the consumables that must be replaced within 500 km in orange, and the consumables that are safe in green. The visualized information is placed on the display inside the vehicle.

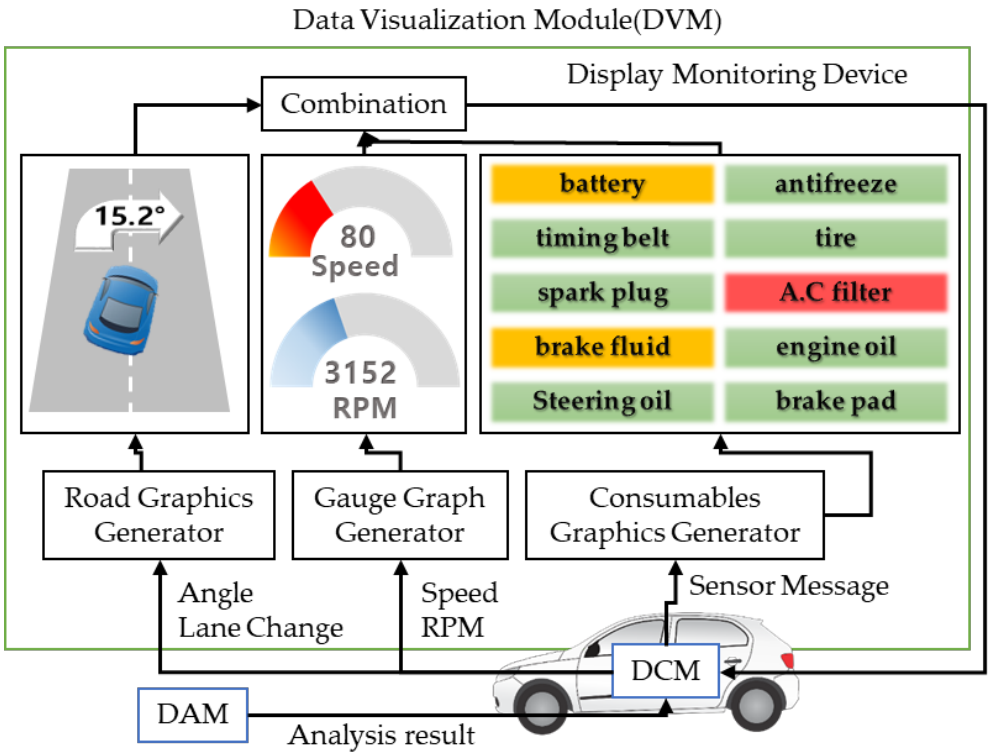


Fig. 9. The process in which the DVM visualizes data

4. Experimental Results and Analysis

In this paper three experiments were conducted to verify the validity of the ODSS. First, to judge whether sensor messages were accurately sent to the Cloud, the DCM was compared with exixting vehicle gateways in the loss rate of message transmission. Second, to judge whether the proposed driving strategy was accurate, the DAM was compared with the Multilayer Perception(MLP) and Random Forest(RF) in the accuracy of the analyzed driving strategies. Third, to measure the real-time of the DAM, the time it takes to determine the RPM, speed, steering angle and lane change was compared to the MLP using the vehicle's sensor data after learning. The experiment was conducted in a virtual environment, and **Table 1** indicates the environment in which the experiment was conducted. Third, the computation time was measured when DAM, MLP and RF decided the optimal driving strategy to measure the real-time of the DAM. The experiment was conducted in a virtual environment and **Table 1** shows the environment in which the experiment was conducted.

Table 1. The experiment environment

CPU	GPU	RAM	OS
Intel i5-7400	Geforce GTX 1050	SAMSUNG DDR4 8GB	Window 10 Education

The DCM was compared with existing vehicle gateways in loss rate when the vehicle transmitted 500 to 5000 CAN messages, FlexRay messages and High Speed CAN (HSCAN) messages from a vehicle to the Cloud separately. Existing gateways had a loss rate of 19% when CAN messages were transmitted to Cloud, 12.7% when FlexRay messages were transmitted to it, and 12.4% when HSCAN messages were transmitted. The DCM had an loss rate of 11.7% when CAN messages were transmitted to Colud, a loss rate of 7% when FlexRay messages were transmitted, and a loss rate of 10.4% when HSCAN messages were transmitted. The average loss rate for existing gateways was 14.7%, while that for the DCM was 9.7%. Therefore, since the DCM improved a loss rate of 5%, compared with existing gateways, it was more suitable for transmitting CAN, FlexRay and HSCAN messages to the Cloud than existing vehicle gateways. **Table 2** shows the experiment result.

Table 2. The experiment result of the DCM and existing gateways in loss rate

The number of messages	Loss rate					
	CAN		FlexRay		HSCAN	
	existing gateway	DCM	existing gateway	DCM	existing gateway	DCM
500	8.0%	7.0%	2.0%	2.0%	9.1%	8.2%
1000	10.8%	9.0%	3.7%	2.7%	8.1%	8.1%
1500	16.3%	7.6%	11.5%	4.5%	7.8%	8.4%
2000	18.9%	10.7%	12.8%	7.2%	9.8%	9.2%
2500	17.7%	7.6%	14.9%	9.5%	10.1%	9.3%
3000	20.5%	11.4%	15.2%	8.9%	10.5%	9.7%
3500	21.8%	12.5%	16.8%	7.8%	14.7%	10.5%
4000	22.5%	13.8%	13.9%	9.2%	16.9%	12.4%
4500	24.5%	16.8%	18.2%	9.1%	17.6%	13.3%
5000	29.5%	20.4%	17.9%	8.7%	19.4%	15.2%

Fig. 10 shows the accuracy of the driving strategies computed by the DAM, MLP and RF. To measure the accuracy of the DAM, the difference between the driving strategies determined by the DAM, MLP and RF is computed as a percentage. The accuracy of the DAM, MLP and RF was measured in determining the 10 driving strategies. The experiment result shows that the DAM has higher accuracy by about 0.3% than the MLP and by about 2.5% than the RF. That is, the accuracy of the DAM does not differ significantly from that of the MLP, but the DAM analyzes the data more accurately than the RF.

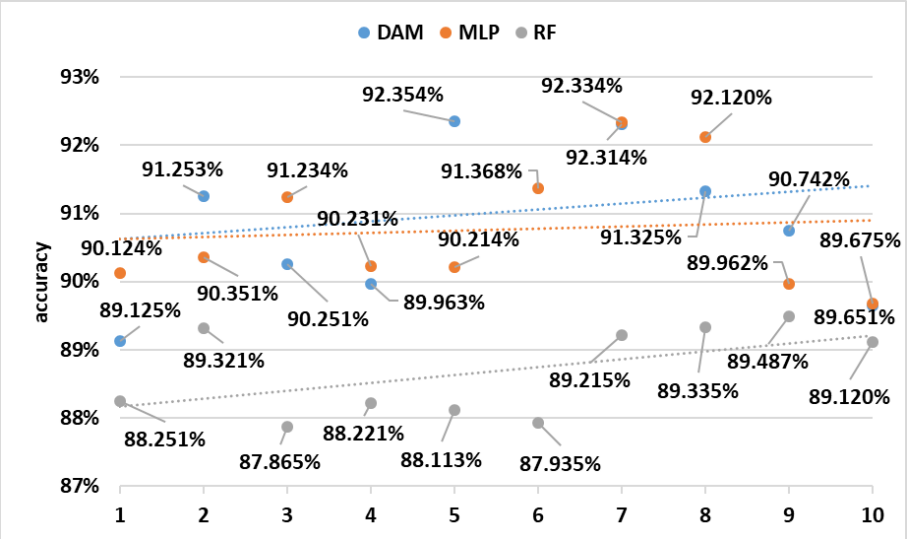


Fig. 10. The comparison of the DAM with MLP and RF in accuracy

Fig. 11 shows the computation time it took for the DAM, MLP and RF to determine the optimal driving strategy when they receive sensor messages from an autonomous vehicle. The computation time of DAM, MLP and RF was measured in determining 10 driving strategies. In the experiment, the DAM determined the optimal driving strategy by about 22% faster than the RF and by 40% faster than the MLP and sent it to an autonomous vehicle. That is, the DAM has an accuracy similar to the MLP, can determine how the vehicle is driving faster, has higher accuracy of 2.5% than the RF and can determine how the vehicle is driving 22% faster.

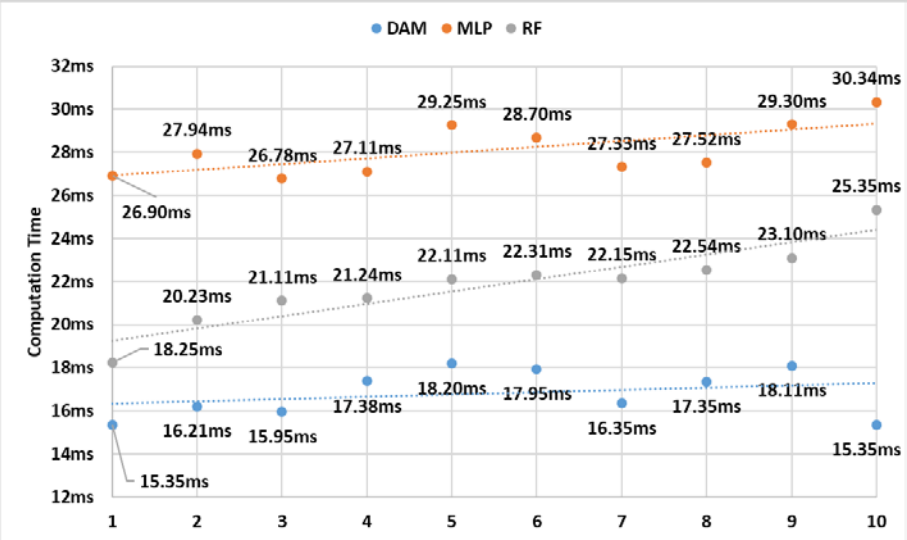


Fig. 11. The comparison of the DAM with MLP and RF in computation time

In conclusion, the ODSS can transmit data by about 10% more accurately than existing in-vehicle gateways, and determine the vehicle's optimal driving strategy faster and more accurately than existing neural network models and real-time machine running methods.

5. Conclusion

This paper proposed Optimal Driving Support Strategy (ODSS). It collects and stores sensor data of vehicles in the cloud, executes the genetic algorithm based on accumulated data to determine the vehicle's optimal driving strategy according to the slope and curvature of the road in which the vehicle is driving and visualizes the driving and consumables conditions of an autonomous vehicle to provide drivers.

To verify the validity of the ODSS, experiments were conducted on the DCM to transmit data from the vehicle to the Cloud and on the DAM to select an optimal driving strategy by analyzing data from an autonomous vehicle. The DCM had a loss rate about 5% lower than the existing vehicle gateways when transmitting CAN messages, FlexRay and HS-CAN messages from the vehicle to the cloud. Though the DAM has a similar accuracy to the MLP, it determines the optimal driving strategy 40% faster than it. And the DAM has a higher accuracy of 22% than RF and determines the optimal driving strategy 20% faster than it. Thus, the ODSS is best suited for determining the optimal driving strategy that requires accuracy and real-time.

The advantages of ODSS proposed in this paper are as follows. First, The ODSS can be used more in a variety of areas in the future because it stores in the Cloud the past data of the safe vehicle driving and accumulates it. Second, because the ODSS sends only the key data needed to determine the vehicle's optimal driving strategy to the cloud and analyzes the data through the genetic algorithm, it determines its optimal driving strategy at a faster rate than existing methods. However, the experiments of the ODSS were conducted in virtual environments using PCs, and there were not enough resources for visualization. Future studies should test the ODSS by applying it to actual vehicles, and enhance the completeness of visualization components through professional designers.

References

[1] SAE International Releases Updated Visual Chart for Its, “Levels of Driving Automation” Standard for Self-Driving Vehicles. https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles

[2] YiNa Jeong, SuRak Son, EunHee Jeong and ByungKwan Lee, “An Integrated Self-Diagnosis System for an Autonomous Vehicle Based on an IoT Gateway and Deep Learning,” *Applied Sciences*, vol. 8, no. 7, pp.1164, july 2018. Article (CrossRef Link).

[3] Hobold Gustavo M, da Silva Alexandre K, “Visualization-based nucleate boiling heat flux quantification using machine learning,” *International Journal of Heat and Mass Transfer*, Vol .134, pp.511-520, 2019. Article (CrossRef Link).

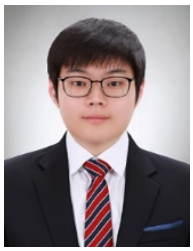
[4] Ning Ye, Yingya Zhang, Ruchuan Wang, Reza Malekian, “Vehicle trajectory prediction based on Hidden Markov Model,” *The KSII Transactions on Internet and Information Systems*, Vol. 10, No. 7, pp. 3150-3170, 2017. Article (CrossRef Link).

[5] Li-Jie Zhao, Tian-You Chai, De-Cheng Yuan, “Selective ensemble extreme learning machine modeling of effluent quality in wastewater treatment plants,” *International Journal of Automation and Computing*, Vol.9, No.6, pp.627-633, 2012. Article (CrossRef Link).

[6] Torben Graber, Stefan Lupberger, Michael Unterreiner, Dieter Schramm. “A Hybrid Approach to Side-Slip Angle Estimation With Recurrent Neural Networks and Kinematic Vehicle Models,” *IEEE Transactions on Intelligent Vehicles*, Vol.4, No.1, pp.39-47, 2018. Article (CrossRef Link).

[7] Yang Zheng, John H. L. Hansen. “Lane-Change Detection From Steering Signal Using Spectral Segmentation and Learning-Based Classification,” *IEEE Transactions on Intelligent Vehicles*, Vol.2, No.1, pp.14-24, 2017. Article (CrossRef Link).

[8] Feng Z. C, Chen J. K, Zhang Y, Griggs J. L, "Estimation of front surface temperature and heat flux of a locally heated plate from distributed sensor data on the back surface," *International Journal of Heat and Mass Transfer*, Vol.54, No. 15-16, pp.3431-3439, 2011. Article (CrossRef Link).
[9] Xiaohui Li, Junfeng Wang, "A Generous Cooperative Routing Protocol for Vehicle-to-Vehicle Networks," *KSII Transactions on Internet and Information Systems*, Vol.10, No.11, pp. 5322-5342, 2011. Article (CrossRef Link).
[10] Ekim Yurtsever, Suguru Yamazaki, Chiyomi Miyajima, Kazuya Takeda, Masataka Mori, Kentarou Hitomi, Masumi Egawa, "Integrating Driving Behavior and Traffic Context Through Signal Symbolization for Data Reduction and Risky Lane Change Detection," *IEEE Transactions on Intelligent Vehicles*, Vol. 3, No. 3, pp.242-253, 2018. Article (CrossRef Link).
[11] Gao Qiang, Zuo Yi, Zhang Jun, Peng Xiao-Hong, "Improving energy efficiency in a wireless sensor network by combining cooperative MIMO with data aggregation," *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 8, pp.3956-3965, 2010. Article (CrossRef Link).
[12] Liang Xue, Xin-Ping Guan, Zhi-Xin Liu, Qing-Chao Zheng, "A power- and coverage-aware clustering scheme for wireless sensor networks," *International Journal of Automation and Computing*, Vo.7, No.4, pp.500-508, 2010., Article (CrossRef Link).
[13] Stephanie Mayer, Gautier Hattenberger, Pascal Brisset, Marius O. Jonassen, Joachim Reuder, "A 'No-Flow-Sensor' Wind Estimation Algorithm for Unmanned Aerial Systems," *International Journal of Micro Air Vehicles*, Vol.4, No.1, pp.15-29, 2012. Article (CrossRef Link).
[14] Yi-Ke Guo, Li Guo, "IC cloud: Enabling compositional cloud," *International Journal of Automation and Computing*, Vol.8, No.3, pp.269-279, 2011.
[15] Bing Li, Bu-Qing Cao, Kun-Mei Wen, Rui-Xuan Li, "Trustworthy assurance of service interoperation in cloud environment," *International Journal of Automation and Computing*, Vol.8, No.3, pp.297-308, 2011. Article (CrossRef Link).
[16] Sheng Liu, Xu-Cheng Chang, "Synchro-control of twin-rudder with cloud model," *International Journal of Automation and Computing*, Vol.9, No.1, pp.98-104, 2012. Article (CrossRef Link).
[17] Yukiko Kenmochi, Lilian Buzer, Akihiro Sugimoto, Ikuko Shimizu, "Discrete plane segmentation and estimation from a point cloud using local geometric patterns," *International Journal of Automation and Computing*, Vol. 5, No. 3, pp.246-256, 2008. Article (CrossRef Link).
[18] Xiaojiang Du, Hongli Zhang, Qiang Zhang, "Toward Vehicle-Assisted Cloud Computing for Smartphones," *IEEE Transactions on Vehicular Technology*, vol.64, No.15, pp.5610-5618, 2015. Article (CrossRef Link).
[19] YiNa Jeong, SuRak Son and ByungKwan Lee, "The Lightweight Autonomous Vehicle Self-Diagnosis (LAVS) Using Machine Learning Based on Sensors and Multi-Protocol IoT Gateway," *Sensors*, vol. 19, no. 11, pp. 2534, June 2019. Article (CrossRef Link).
[20] YiNa Jeong, SuRak Son, SangSik Lee and ByungKwan Lee, "A Total Crop-Diagnosis Platform Based on Deep Learning Models in a Natural Nutrient Environment," *Applied Sciences*, vol. 8, no. 10, pp. 1992, October 2018. Article (CrossRef Link).



SuRak Son received his B.Eng degree from Kwandong University in 2018. He is currently working towards a master in Computer Science from Catholic Kwandong University. Her current research interests are Artificial neural network, Autonomous vehicle, and Network security. Email: sonsur@naver.com

YiNa Jeong received her B.Eng. degree from Kwandong University in 2011, the M.Eng. degree in Computer Science from Kwan-Dong University in 2012. She is currently working towards a doctorate in Computer Science from Catholic Kwandong University. Her current research interests are Sensor Network, IT security, and Network security Email: upinus07@nate.com

ByungKwan Lee received his B.S. degree from Pusan National University in 1979, the M.S. degree in Computer Science from Chung-Ang University in 1986 and the ph.D. degree in Computer Science from Chung-Ang University in 1990 in Korea. He has been a professor of Computer Science at Catholic Kwandong University in Korea since 1988. He was a visiting professor at Saginaw Valley State university, Michigan, USA during 2000~2001. He is a permanent member of the KISS and KIPS. His current research interests are network security, Wireless Sensor Network Security, Internet of Things and Big Data. Email: bklee@cku.ac.kr