공학박사학위논문

자율 주행 시스템의 차량 안전을 위한 적응형 관심 영역 기반 효율적 환경 인지

Efficient Environment Perception based on Adaptive ROI for Vehicle Safety of Automated Driving Systems

2020년 2월

서울대학교 대학원

기계항공공학부

박 성 렬

Abstract

Efficient Environment Perception based on Adaptive ROI for Vehicle Safety of Automated Driving Systems

Sungyoul Park

School of Mechanical and Aerospace Engineering

The Graduate School

Seoul National University

Since annually 1.2 million people die from car crashes worldwide, discussions about fundamental preventive measures for traffic accidents are taking place. According to the statistical survey, 94 percent of all traffic accidents are caused by human error. From the perspective of securing road safety, automated driving technology became interesting as a way to solve this serious problem, and its commercialization was considered through a step-by-step application through research and development. Major carmakers already have developed and commercialized advanced driver assistance systems (ADAS), such as lane keeping assistance system (LKAS), adaptive cruise control (ACC), parking assistance system (PAS), automated emergency braking (AEB), and so on. Furthermore, partially automated driving systems are being installed in vehicles and released by carmakers. Audi AI Traffic Jam Pilot (Audi), Autopilot (Tesla), Distronic Plus (Mercedes-Benz), Highway Driving Assist (Hyundai Motor Company), and Driving Assistant Plus (BMW) are typical released examples of the partially automated driving system. These released partially automated driving systems are still must be accompanied by driver attention. Nevertheless, it is proving to be effective in significantly improving safety.

In recent years, several automated driving accidents have occurred, and the frequency is rapidly increasing and attracting social attention. Since vehicle accidents are directly related to human casualty, accidents of automated vehicles cause social insecurity by causing a decrease in the reliability of automated driving technology. Due to recent automated driving-related accidents, the safety of the automated vehicle has been emphasized more. Therefore, in this study, we propose an approach to secure vehicle safety in terms of the entire system in consideration of the behavior control of the automated driving vehicle.

In addition, the development of automated driving is not merely a replacement technology for driving, but it is expected to have an industrial assembly as integration of high technology. Currently, automated driving systems have been extended from the conventional framework of the existing automotive industry, and are being developed in various fields. Since automated driving is composed of a complex combination of various technologies, development is currently underway in various conditions and has not been standardized yet. Most developments tend to pursue local performance improvement in each module unit, and the overall system unit approaches considering the relationship between component modules is insufficient. Local research and development at the submodule level can be challenging to achieve adequate performance from a system-level due to the effects of module interaction in terms of system integration perspective. The one-way approach that considers only the performance of each module has its limitations. To overcome this problem, it is necessary to consider the characteristics of the modules involved.

This dissertation focuses on developing an efficient environment perception algorithm by considering the interaction between configured modules in terms of entire system operation to secure the stable and high performance of an automated driving system. In order to perform effective information processing and secure vehicle safety from a practical perspective, we propose an adaptive ROI based computational load management strategy. The motion characteristics of the subject vehicle, road design standards, and driving tasks of the surrounding vehicles, such as overtaking, and lane change, are reflected in the design of adaptive ROI, and the expansion of the area according to the

driving task is considered. Additionally, motion planning results for automated driving are considered in the ROI design in order to guarantee the practical safety of the automated vehicle. In order to secure reasonable and appropriate environment information for the wider areas, lidar sensor data is classified by the designed ROI, and separated processing is conducted according to area importance. Based on the driving data, the calculation time of each module constituting the target system is statistically analyzed. In consideration of the system performance constraint determined by using human reaction time and industry standards, target hardware specification and the performance of sensor, the appropriate sampling time for automated driving system is defined to enhance safety. The data-based multiple linear regression is applied to predict the computation time by each function constituting perception module, and the computational load reduction is applied sequentially by selecting the data essential for automated driving safety based on adaptive ROI to secure the stable real-time execution performance of the system. In computational load assessment, it evaluates whether the computational load of the environmental perception module and entire system are appropriate and restricts the vehicle behavior when there is a problem in the computational load management to ensure vehicle safety by maintaining system stability.

The performance of the proposed strategy and algorithms is evaluated through driving data-based simulation and actual vehicle tests. Test results show that the proposed environment recognition algorithm, which considers the interactions between the modules that make up the automated driving system, guarantees the safety of automated vehicle and reliable performance of system in an urban environment scenario.

Keywords: Automated driving system, Adaptive ROI, Lidar processing, Environment perception, Vehicle safety, Computational load management

Student Number: 2014-22484

List of Figures

Contents

Chapter 1 Introduction

1.1. Background and Motivation

Worldwide, 1.2 million people are killed in car accidents every year, with the number of road traffic deaths rising steadily [WHO,'15]. There were 37,461 people killed in crashes on U.S. roadways during 2016, a 5.6% increase over the previous year [NHTSA,'17]. Besides, road traffic injuries are the eighth leading cause of death for all age groups. More people died from road traffic injuries than from HIV/AIDS, tuberculosis, or diarrhoeal diseases [WHO,'18]. As the increase in traffic accidents on roads causes serious social problems, then discussions about fundamental preventive measures for traffic accidents are taking place.

According to the statistical survey from national highway traffic safety administration (NHTSA) in the U.S., 94 percent of all traffic accidents are caused by human error. Among the accidents caused by human error, the recognition errors accounted for 41 percent (±2.1%), decision errors 33 percent (±3.7%), and the performance errors 11 percent (±2.7%) of the crashes [Singh,'15b]. Moreover, during many types of collision accidents, most drivers do not attempt to avoid crashes due to unawareness of collision risks [Tideman,'07]. Furthermore, it is estimated that 5~35 percent of all road deaths are reported as alcohol-related [WHO,'18].

From the perspective of securing road safety, automated driving technology

became attractive as a way to solve this problem, and its commercialization was considered through a step-by-step application through research and development. Major carmakers already have developed and commercialized advanced driver assistance systems (ADAS), such as lane keeping assistance system (LKAS), lane change assistance (LCA), adaptive cruise control (ACC), parking assistance system (PAS), automated emergency braking (AEB), vehicle stability control (VSC), blind spot intervention (BSI) and so forth. [Hoedemaeker,'98, Bishop,'00, Tingvall,'00, Kato,'02, Netto,'04, Tideman,'07, Naranjo,'08, Moon,'09, Gordon,'10, Kastner,'11, Zhang,'11].

Furthermore, partially automated driving systems have been released by carmakers. Audi AI Traffic Jam Pilot (Audi), Autopilot (Tesla), Distronic Plus (Mercedes-Benz), and Highway Driving Assist (Hyundai Motor Company), and Driving Assistant Plus (BMW) are typical released examples of the partially automated driving system [Brenner,'18]. These released partially self-driving systems are still must be accompanied by driver attention. Nevertheless, it is proving to be effective in significantly improving safety.

The primary issue in terms of the advancement and commercialization of automated driving is the accurate and rapid recognition performance of the surrounding environment and rational decision based on it. Most of the self-driving car accidents in recent years have proved to be cognitive performance problems, which is the reason why they support them. To improve environment perception performance, it may be the ultimate solution to mount many sensors and to fuse much information quickly, but there are practical limitations in terms of cost. In terms of the commercialization of automated vehicles, it is

necessary to consider the hardware capability and cost realistically. Therefore, an effective and efficient approach is required in the process of applying the sensor for providing high-precision environment information in addition to the sensor already in mass production.

This study focuses on the lidar sensor, which is emerging as the core of environment awareness for automated driving in the current sensor technology level. Lidar is a product of high-level optical technology and can improve recognition performance with higher accuracy and precision compared to conventional environment sensors. Conventional cameras and radars have sufficient sensing capabilities for the purpose of applying ADAS, but not enough to be applied to automated driving systems of level 3 and above. Lidar is currently less practical in terms of production cost because it is difficult to mass-produce. It was assessed that lidar is challenging to commercialize because it requires considerable computational resources. However, due to advances in technology, the price of lidar has gradually dropped, and although still classified as an expensive sensor, it can be installed in the Audi A8 2018 model [Zhao,'19]. Although it is currently used to provide environmental sensing for partial front areas due to cost issues, the coverage and number of applications are expected to increase gradually. Therefore, perception technology using lidar is not only a medium-to-long-term prior technology but also needs to be applied as a technology to be applied in the present and near future.

As technology advances, more and more data can be obtained with sensors, the amount of computation increases exponentially. Since point cloud

processing is performed in consideration of the correlation between each point, as the number increases, the computational complexity can also be considered proportional [Asvadi,'16]. Because the amount of data and the correlation between the data vary depending on the sensing situation, there is a considerable variation in the data size that must be processed every moment. It causes a considerable variation in the computation. It can be solved simply by securing high-performance hardware that can stably handle the most substantial amount of data that can be acquired, but this is not a suitable solution due to the high cost. Thus, a realistic approach is needed to solve this problem through optimization in the system operating environment. For the perception algorithm to run properly, the variation in the amount of data to be processed must be reasonably selected and reduced, even with changes in the sensed environment.

In addition, the development of automated driving is not simply a replacement technology for driving, but it is expected to have a industrial assembly as an integration of high technology. Currently, automated driving systems have been extended from the conventional framework of the existing automotive industry, and are being developed in various fields. Since automated driving is composed of a complex combination of various technologies, development is currently underway in various different conditions and has not been standardized yet. Most developments tend to pursue local performance improvement in each module unit, and the overall system unit approach considering the relationship between component modules is insufficient. Local research and development at the submodule level can be difficult to achieve adequate performance from a system level due to the effects of module

interaction in terms of system integration perspective. The one-way approach that considers only the performance of each module has its limitations. To overcome this problem, it is necessary to consider the characteristics of the modules involved.

Therefore, this dissertation focuses on developing an efficient perception algorithm to secure the stable performance and vehicle safety of an automated driving system in terms of the practical point of view, which gradually develops from partially automated driving to fully automated driving technology development.

1.2. Previous Researches

A number of studies have been introduced for the development of an automated driving algorithm. Zhu et al. designed control system for parking of an automated vehicle and verified by implementing it on a truck to conduct demonstration [Zhu,'06]. Kim et al. proposed a fully automated driving algorithm on complex urban roads with lidar, vision, and GPS/map based environment representation with guaranteed safety [Kim,'15a]. Jo et al. applies distributed system architecture to the autonomous driving system, to obtain reduction of the computational complexity of the entire system, fault-tolerant characteristics, and modularity of the system [Jo,'14]. Bertha Benz proposed vison and radar-based perception, digital road maps and video-based self-localization, as well as motion planning in complex urban scenarios and verified through vehicle test in fully autonomous manner equipped with close-to-production sensor hardware [Ziegler,'14]

Comprehensive and precise environment perception is the basis for safe and comfortable autonomous driving in urban complex situations [Vanholme,'13]. To improve environment recognition performance, various sensors such as radar [Hutchison,'10, Ziegler,'14, Giese,'17], lidar [Börcs,'17, Magnier,'17, Moras,'11, Salti,'14], monocular vision [Premebida,'07, Hadsell,'09, Sivaraman,'13, Ren,'15], stereo vision [Bertozzi,'00, Kaempchen,'02, Oniga,'10, Li,'18], ultrasonic sensor [Satonaka,'06, Adarsh,'16], around view monitoring (AVM) camera [Jo,'15, Kim,'16, Park,'16, Lee,'17] have been studied to

recognize the environment situation for ADAS or automated driving system.

Among the environment sensors for automated driving systems, lidar plays the most crucial role in high-level automated driving due to the high resolution and accuracy of distance. Also, lidar has been widely used for high-definition map (HD Map) construction and map-based localization [Bosse,'09, Wolcott,'14, Hata,'14a, Hata,'14b], simultaneous localization and mapping (SLAM) [Hess,'16], detection and tracking of moving objects (DATMO) and object recognition [Asvadi,'16, Wojke,'12, Feng,'18, Gao,'18]. Fuerstenberg et al. proposed pedestrian detection algorithm to improve object tracking and classification performance by considering the distance and reflectivity of lidar [Fuerstenberg,'04]. A lot of studies of model based object tracking algorithms using lidar sensor have been proposed [Mendes,'04, Ye,'16, Cho,'14, Zhang,'17].

The huge volumes and complexity of lidar data are to be significant challenges for data processing as the limitation of the computing hardware. With conventional sequence algorithms, massive point cloud processing is to be time-consuming because the processing is computationally intensive and iterative [Yang,'13, Wu,'11, Asvadi,'16]. Thus, the development of alternative solutions is urgently needed in practical applications. Various optimization techniques and algorithms have been proposed to improve the performance of lidar point cloud processing [Elseberg,'11, Isenburg,'06, Han,'09]. Some of those, parallel processing, is to be a potential lidar processing solution [Liu,'12, Będkowski,'13]. Cao et al. proposed a data processing structure by integrating parallel computing based on efficient network topology to improve the processing efficiency [Cao,'15]. However, these approaches are typically

designed about the parallel architecture of target systems such as multi-core processors, GPU, etc. These approaches show better performance indicators than the conventional processing approach, but it is not practical from the commercialization point of view since it still requires high hardware performance.

Due to the significant computational load for vision-processing algorithms for high-resolution images, various studies have been conducted to simplify or minimize computations. Benligiray et al. proposed a simple and video-based lane detection algorithm that uses a fast-vanishing point estimation method in order to obtain real-time performance. The angle-based elimination of the line segment reduced the number of features to be processed afterward to make execution time for each frame stable [Benligiray,'12]. Ding et al. proposed a vision-based road ROI determination algorithm to detect efficiently road region using the positional information of a vanishing point and line segment. Road ROI was first detected, and processing was performed at such determined ROI, which improved recognition accuracy and calculation efficiency [Ding,'13]. Baek et al. developed an efficient algorithm to set adaptive ROI for detecting pedestrians in a moving vehicle in order to reduce computation time and maintain the performance of the conventional method [Baek,'12]. Samejima et al. proposed the autonomous adaptive ROI selection method with a risk evaluation of the working condition by an autonomous monitoring robot. Autonomous ROI selection is realized by the relationship evaluation based on the gestalt factor. Experiment results confirmed that the reduction of working time and the number of the concurrence of the error [Samejima,'16]. Grois et al.

proposed a method and system for the scalable video coding by presenting a complexity-aware adaptive spatial ROI SVC pre-filtering scheme. The ROI SVC visual presentation quality is significantly improved, which can be especially useful for various resource-limited devices in real-time [Grois,'11]. In many vision-processing studies above, execution time was reduced by performing a selective operation on variable ROI. This concept of the approach is applicable to lidar processing for environment perception in a similar way.

Computing huge amounts of data every cycle is very inefficient in terms of overall system resource management, and is uneconomical because it requires high-end hardware to ensure performance. In various research fields, computational load problem has become an important issue from a practical point of view. Among the various methods, the key-frame concept that selects and applies the currently necessary data according to the conditions has been studied on various topics. Mouragnon et al. presented a application of SFM techniques to localization and mapping, for a moving car. Their model is built in real-time with 3D points reconstructed from interest points extracted in images and matched through the monocular video sequence [Mouragnon,'06]. Kim et al. proposed a robust loop detection method by matching image features between the incoming image and key-frame images saved in SLAM [Kim,'07]. In text detection, a method has been proposed to efficiently extract the key frames from the videos based on color moments and then text localization is done only on the key frame [Singh,'15a]. In this way, the key frame approach to increase the computational efficiency is effective from a practical point of view. It is expected that the concept of key frames of images and maps can be

similarly applied as ROI concepts to point cloud data processing.

A prediction of the execution time of computer programs is an important but challenging problem in the community of computer systems. Iverson et al. proposed a statistical execution time estimation algorithm for use in a heterogeneous distributed computing environment. This algorithm makes predictions using past observations of the execution time. These estimates compensate for the properties of the input data set and the machine type, without requiring any direct knowledge of the internal operation of the task or machine [Iverson,'99]. Huang et al. proposed Sparse POlynomial REgression (SPORE) algorithms that use the automatically extracted features to predict a computer program's performance using feature data collected from program execution on sample inputs [Huang,'00]. Yamamoto et al. propose an execution time prediction method that combines measurement-based execution time analysis and simulation-based memory access analysis. They used a measurement of basic block execution times on a real machine [Yamamoto,'06]. Regarding these studies, the predicting execution time method through structural algorithm analysis and actual execution result data can be applied to lidar processing.

From a considerable amount of literature above, it is possible that various methods and concepts can be effectively introduced to develop the proposed environment perception algorithm.

1.3. Thesis Objectives

This dissertation focuses on developing an efficient environment perception algorithm by considering the interaction between configured modules in terms of entire system operation to secure the stable and high performance of an automated driving system. In order to perform effective information processing and secure vehicle safety from a practical perspective, we propose an adaptive ROI based computational load management strategy. The motion characteristics of the subject vehicle, road design standards, and driving tasks of the surrounding vehicles, such as overtaking, and lane change, are reflected in the design of adaptive ROI, and the expansion of the area according to the driving task of the vehicle is considered. Additionally, motion planning results for automated driving are considered in the ROI design in order to guarantee the safety of the automated vehicle. To secure reasonable and appropriate environment information for the broader areas, point cloud data is classified by the designed ROI, and separated processing is conducted according to area importance. Based on the driving data, the calculation time of each module of the target system is statistically analyzed. In consideration of the system performance constraint determined by using human reaction time and industry standards, target hardware specification and the performance of sensor, the appropriate sampling time for automated driving system is determined to enhance safety. The data-based multiple linear regression is applied to the perception module in order to predict the computation time by each function,

and utilize it to secure the reliable real-time performance of the system by applying computational load reduction in stages. In addition, it evaluates whether the computational load of the environmental perception module is appropriate and restricts the vehicle behavior when there is a problem in the computational load management to ensure vehicle safety by maintaining system safety.

Mainly three research issues are considered: processing based on adaptive ROI, environment perception, and computational load management. In the remainder of this thesis, we will provide an overview of the overall architecture of the proposed perception algorithm for automated driving and the performance evaluation based on experimental results, which show the effectiveness of the proposed algorithm. The proposed environment perception algorithm is evaluated through data-based simulation and actual vehicle tests. Test results show that the proposed environment awareness algorithm, which considers the interactions between the modules that make up the system, guarantees the safety of automated vehicle and reliable performance of system in an urban environment scenario.

1.4. Thesis Outline

This dissertation is structured in the following manner. The overall architectures of automated driving and the proposed perception strategy are described in Chapter 2. Chapter 3 presents the design of adaptive ROI and separated processing algorithms by ROI. The primary purpose of adaptive ROI is to properly define the necessary detection areas based on vehicle status and result of motion planning to ensure driving safety by importance and process the categorized data by area to achieve both computational efficiency and accuracy. In Chapter 4, the environment perception algorithms are introduced. The perception algorithms consist of a sensor delay analysis and compensation, static obstacle map construction, detection of road facilities, and moving object tracking and estimation. In Chapter 5, the adaptive ROI based computational load management in order to prevent the execution timeout of the environment perception module is described. The processing time estimation of significant environment recognition functions is designed based on multiple linear regression by using driving data. Then the processing computational load management strategy including processing time reduction by applying sequential processing and restriction of driving condition to achieve real-time computational reliability has also been proposed and validated. Chapter 6 presents the vehicle experiment results to evaluate the performance. Then the conclusion which consists the summary and contribution of the proposed algorithm and future works is presented in Chapter 7.

Chapter 2 Overall Architecture

2.1. Automated Driving Architecture

The overall architecture of our automated driving system is outlined in Figure 2.1. For an automated driving system, mainly three research issues are considered: an environment representation, motion planning, and vehicle control. Environment representation consists of environment perception and localization. The environment perception computes boundary and state information of surrounding objects by processing data obtained from the sensor, and the localization module conducts global positioning of ego vehicle using GPS, inertial sensor, and environment information. The results of environment recognition and vehicle positioning have a direct impact on the performance of motion planning for appropriate driving. The objective of the motion planning modules is to derive an optimal path as a function of time by utilizing the environment representation results. Based on the environment representation results from the perception module, the moving objects are classified, and behavior prediction is performed according to the characteristics of the classified objects. All environment information is represented on the same plane and is used to redefine the drivable corridor from the initial guess. The desired longitudinal acceleration and desired path are determined using the Model Predictive Control (MPC) approach. Safety, dynamics, and actuator

constraints are simultaneously considered to optimize the desired motion of the vehicle. The vehicle control module feeds back the pose estimate of the localization module to guide the vehicle along the planned trajectory.

This study focuses on the environment perception algorithm. The developed algorithm in this study is implemented in the automated driving system of Figure 2.1 to perform its role. Figure 2.1 summarizes the main functions, such as adaptive ROI based processing, computation load management, and environment representation proposed in this study. The detailed structure of adaptive ROI based perception can be seen in Figure 2.2.

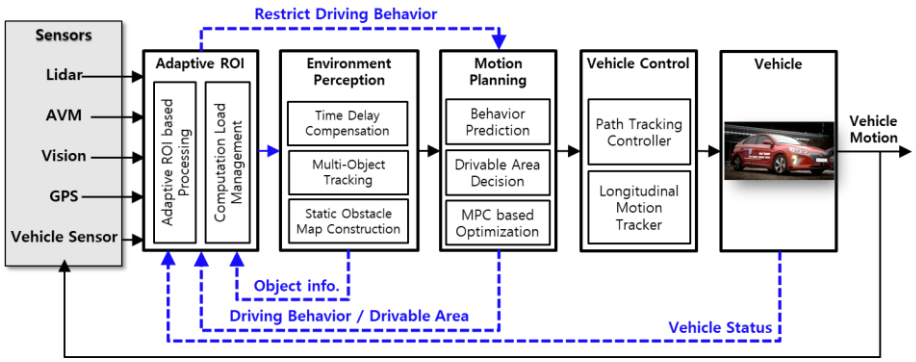


Figure 2.1. Overall architecture of automated driving system.

The proposed algorithm consists of the following three stages: regional processing, computation load management, and environment perception, as described in Figure 2.2 by colored blocks. The algorithm proposed in this study aims at maximizing recognition ability and execution stability by strategically performing massive data processing through rational ROI design in terms of entire system perspective.

In the first stage, the adaptive ROI definition designs an adaptive ROI for normal driving, lane change, and intersection passing situations. By using lane geometry obtained from vehicle status information, road design standards, and vision, the ROI is designed to be flexible and adapt to every situation. Adaptive ROI-based separated data processing is then performed. Point cloud classification is the most basic classification process for classifying data by region. The point clouds classified by ROI are downsizing and clustering according to ROI importance. Unlike conventional methods that apply uniformly to the entire data, parameter settings are applied differently for each important area. As a result, processing performance is maximized while minimizing distortion or loss of data.

Nevertheless, depending on the operating conditions of the system and the surrounding environment, there is still a possibility of exceeding the time limit due to the computational load. Thus, we propose a computational load assessment method to prevent such a failure. The computational load management consists of processing time prediction, sequential processing, and restriction of driving conditions. In processing time prediction, multiple linear regression is applied to estimate the computation time using real driving data. We select a model that properly reflects algorithm computation characteristics and system operating environment characteristics. Using the determined processing time prediction model for each function, the algorithm determines whether the algorithm timeout before processing and applies a strategy to prevent it. Also, to ensure the minimum driving safety of the automated driving

system, a restriction function that partially limits the driving task or reduces the top speed is applied.

In Figure 2.3, the process flow of the computational load management strategy is classified into three colored states. First, it is not necessary to reduce the computational load because it is determined that the normal operating state shown in green is sufficient to process all the data obtained through adaptive ROI-based processing. The computation load reducing state in the blue line indicates a process in which computational load reduction through stepwise ROI is applied because the computational load exceeds the allocated resources. Finally, the restricting driving behavior state, which is marked in red, represents an extreme situation in which the function is restricted by controlling the vehicle behavior in order to secure system stability by reducing the system computational load. In most cases, the system operates with the first and second states, and the third state is activated when there is not enough computational load reduction intermittently.

The proposed environment perception stages consist of an analysis of the characteristics of external environment sensors that are widely used in automated driving systems, the environment perception algorithms needed for the motion planning and control process by processing signals, and the computational load management to ensure real-time reliability of environment recognition systems for automated driving control.

The stages outlined in this section are described in detail in Chapter 3, Chapter 4, and Chapter 5, respectively.

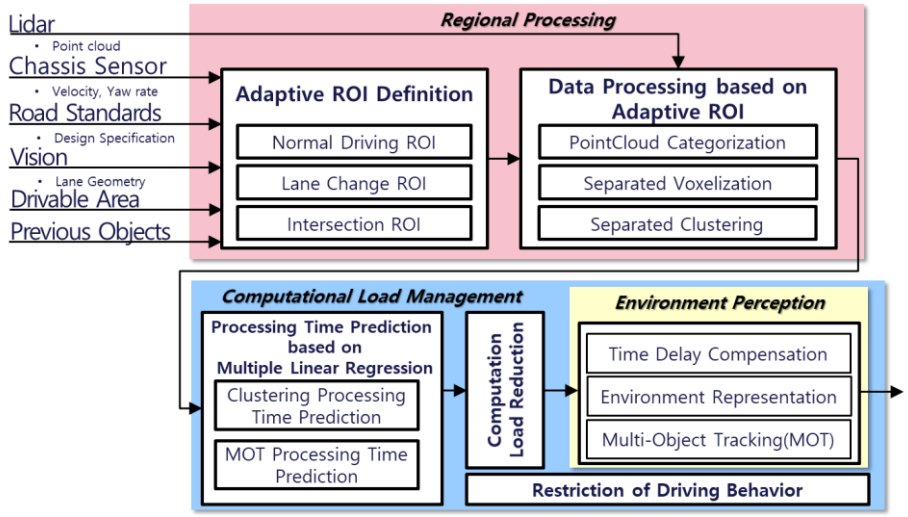


Figure 2.2. Proposed architecture of adaptive ROI based environment perception with computational load management.

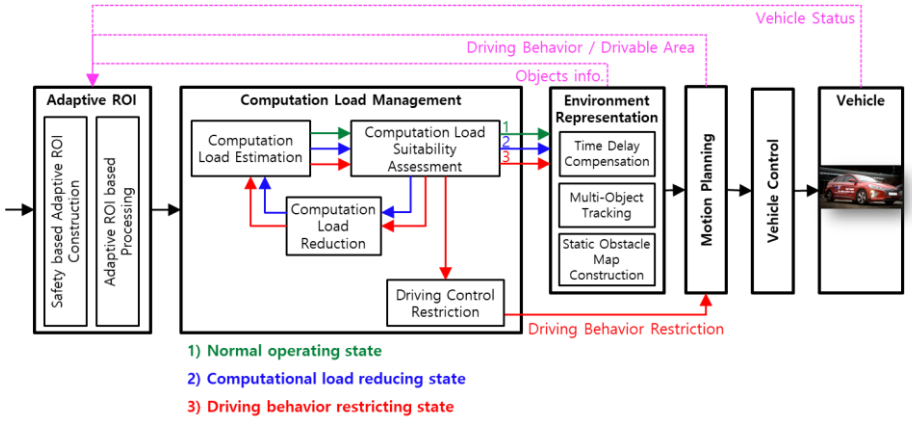


Figure 2.3. Process flow of computational load management

2.2. Test Vehicle Configuration

Automated driving in complex situations, such as urban environments, requires very accurate, precise, and rapid environment perception ability. In a motorway environment, it is considered a relatively simple environment because only vehicles that drive in the same direction as the subject vehicle and road facilities are considered. On the other hand, in the urban environment, various types of roads such as intersections, crosswalks, and roundabouts are constructed in various forms and various objects such as pedestrians, motorcycles, bicycles, and other road facilities must be recognized. Therefore, it is considered to be relatively complicated and challenging from a technical point of view. The development of fully autonomous driving technology is the ultimate goal, but if you make up your system with high-performance, high-performance sensors to achieve this, the gap between reality becomes quite large. For this reason, we aim to develop systems that realistically consider both mass production and high performance. Therefore, we focus on solving the main issues in the process of developing from the already commercialized ADAS, partially automated driving system to the future fully autonomous driving system.

The complete sensor setup for automated driving is shown in Figure 2.4. Six multilayer lidar for environment detection are depicted in red and monocular front vision is marked in yellow. A low-cost GPS is shown in green color and around view monitoring (AVM) cameras are depicted in blue. Computers,

controllers and other major equipment are also briefly described. In order to ensure high perception performance and to realize a high level of autonomous driving with safe and comfort, various sensors and equipment are installed on the test vehicle.

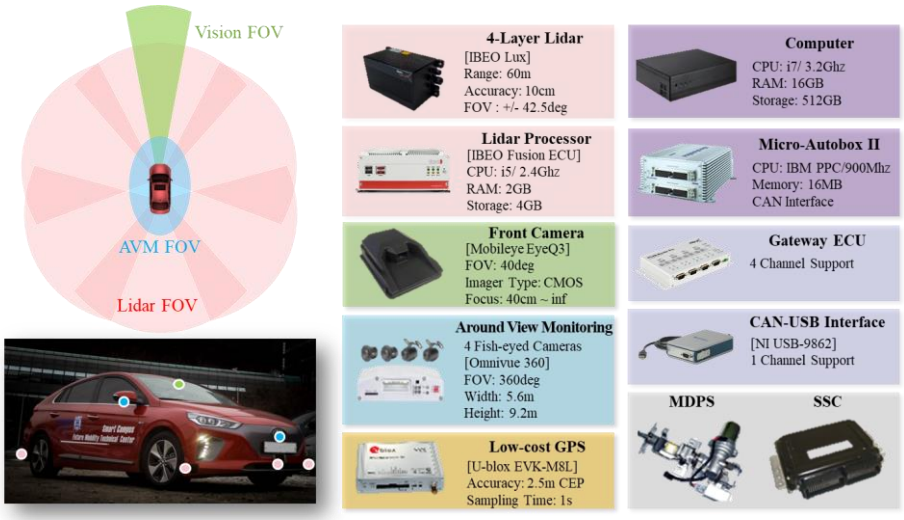


Figure 2.4. The experimental vehicle and its sensor-setup.

Chapter 3 Design of Adaptive ROI and Processing

In order to improve the performance of the environment perception algorithm and ensure the safety of automated vehicle from a practical point of view, we propose an adaptive ROI method considering interaction of configured modules for automated driving using the differential operation method according to the real-time ROI concept. To define the necessary detection area that must be guaranteed for safe driving, the subject vehicle's driving status information, recognized lane information, road design criteria, and vehicle's situational characteristics are used. In addition, motion planning and control results of previous step are considered for the ROI design that can guarantee the safety of the automated vehicle. Moreover, the voxelization and clustering process, which is essential for estimating the state variables of surrounding objects, has been improved to be performed according to the designed adaptive ROI.

Before designing ROI, this chapter defines the self-driving mode that considers the active function and environment of an automated driving system and reflects it in the design. The level of automated driving considering functions and the environment was classified in the previous Chapter 2. Figure 3.1 shows typical driving situations for each level. In addition, the concept is shown in Figure 3.2 by dividing the essential cognitive areas by importance in consideration of the main factors of the typical driving situation of each mode in Figure 3.1. This type of ROI design will be discussed in detail in Section 3.1.

First, at the basic level, the vehicle is determined to travel, and motion planning and control are required to avoid collisions in the driving path. The vehicle generally performs the road keeping function and the ACC function for the front object under the assumption that the vehicle is traveling on the road to which the road traffic law is applied. Second, for motorways, the action of changing lanes in the aforementioned Basic level is added. Lane change requires a more significant amount of steering control than steering control for lane keeping, and additional deceleration and acceleration control is also required depending on the decision and strategy to change lanes. Therefore, these factors are considered in expanding the required recognition range when changing lanes. Finally, urban environments not only need to detect non-vehicle objects such as pedestrians and bikes but also require cognitive performance in geometrically complex environments such as intersections. The urban mode requires coping with various objects and situations in spite of the relatively low speed compared to the motorway mode. Therefore, this study aims to secure perception performance through additional ROI design, and this study deals with a representative intersection environment in an urban environment.

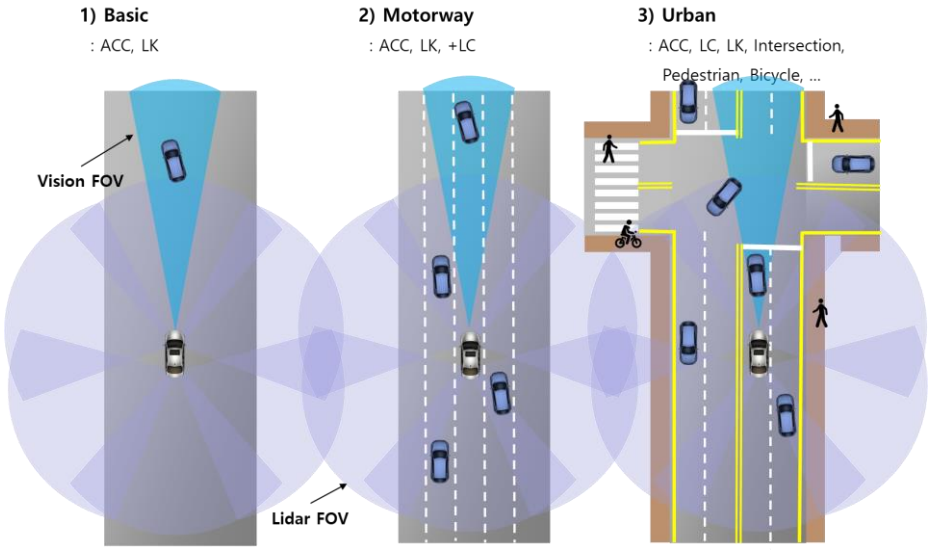


Figure 3.1. Characteristics of typical driving environment.

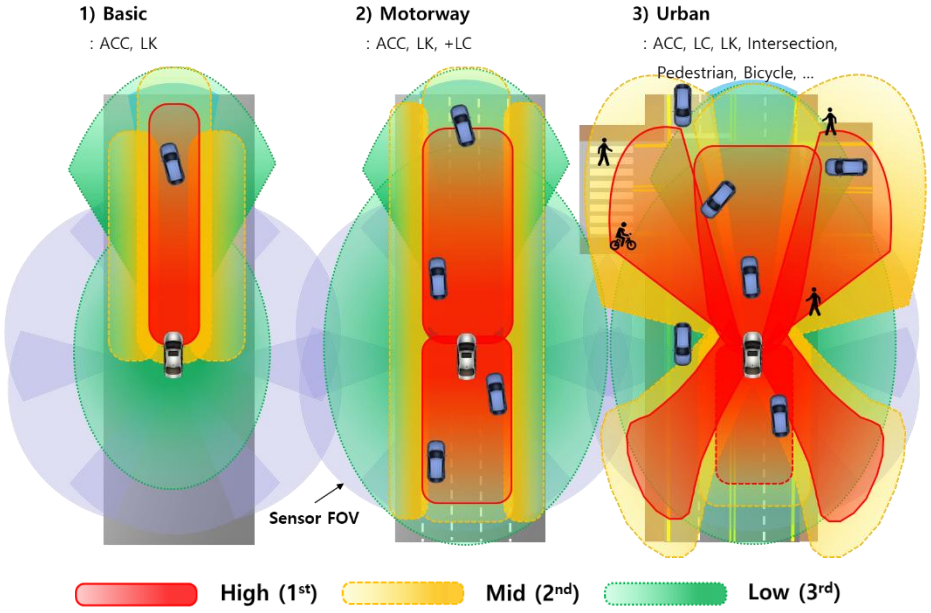


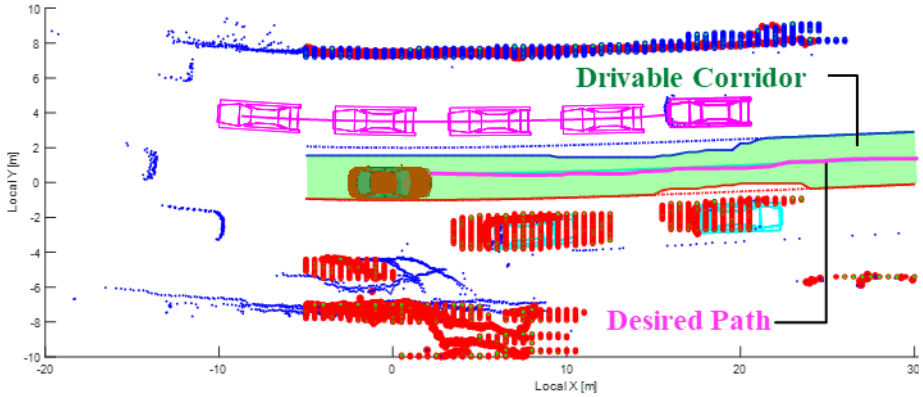
Figure 3.2. Perception ROI scheme on typical driving environment and classified ROI by importance.

Based on the premise that self-driving cars drive on roads that are designed to meet standards, they can define areas that need recognition by using road design standards and information on driving conditions of automated driving vehicles. The behavior of surrounding vehicles in each area is taken into account in the ROI definition. Besides, forward lane information obtained from the front vision sensor is used in the region of interest design. To guarantee safety for automated vehicle, motion planning and control results are highly considered in ROI design. Since the ROI is constructed and applied in real-time in consideration of various information and the result of automated driving control, the designed area is called an adaptive ROI. Depending on the importance, it is designed into three levels. The computational load management proposed in Chapter 5 also uses this ROI of importance. Based on the designed ROI for each level, the point cloud is categorized, voxelized, and clustered to improve computational efficiency and perceptive accuracy.

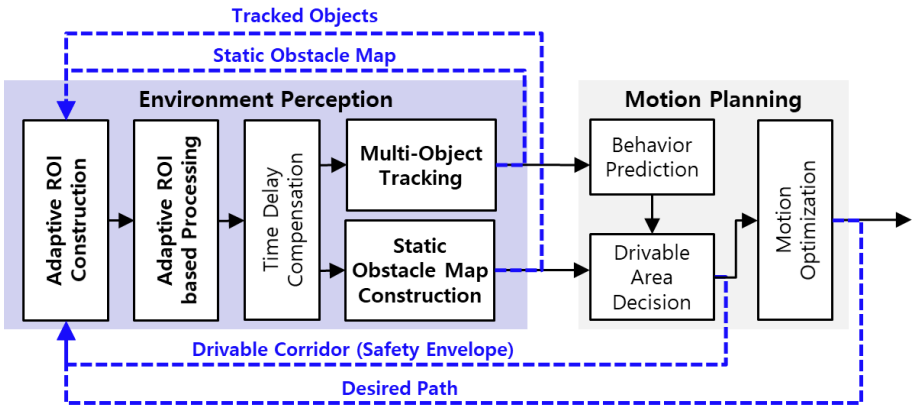
The performance of the proposed adaptive ROI based regional processing is verified via data-based simulation by comparison with the conventional processing approach. The comparison of performance is conducted under the same hardware environment using driving logs, including vehicle status, data of environment sensors, and so forth. It is proven that the proposed approach reduced computation load against typical approaches to ensure efficiency and effectively increase environment recognition performance.

3.1. ROI Definition

In this section, the region of interest for differential processing for each region is defined to efficiently process a large amount of raw data acquired from lidar. The ROI proposed in this section is not fixed and is designed flexibly by considering every situation based on driving condition information, road design standards, and detected lane information. In addition, the motion planning results calculated by the automated driving system are used in the ROI design, thereby increasing the practical safety of the automated vehicle. The motion planning of automated driving system to which this study is applied calculates the desired path based on drivable corridor as depicted in Figure 3.3(a). The driving corridor is determined using the environment representation result and lane information. The optimal desired route is calculated to drive safely inside the driving corridor. In this study, we apply them to ROI design to give priority to the areas that must be recognized for safe automated driving control. Moreover, in order to effectively detect objects that are likely to be considered in motion planning, the previous recognition results are also utilized in the ROI construction. This structure can be seen in Figure 3.3(b).



(a) Drivable area and desired path computed from motion planning



(b) Adaptive ROI construction structure related with other modules

Figure 3.3. Scheme of adaptive ROI design with other modules configuring ADS.

Although the world's specifications are not the same in every country in the world, this does not significantly affect the way people drive in other countries. Similarly, road design standards in each country are not very different in that respect. The highway design criteria for major countries around the world is shown in Table 1.

Although detailed criteria are not consistent among countries, overall, design speed, a width of roadway, and width of lane painting are similar. In this study, we design an adaptive region of interest based on detailed specifications related to road design standards in Korea. Table 2 shows the design speed according to the functional classification of roads announced by the Ministry of Land, Infrastructure, and Transport. Most road infrastructures officially established by the government reasonably design ROI using design parameters such as road width, curvature, and intersection angle. The details related to the road standards used in this study are described in the subsections that describe the specific ROI design process.

Table 1. Road design standards by global country [Ministry of Land,'15].

Country and name of guidelines or other source	Design Speed or Reference Speed [km/h]	Width of Traffic Lane [m]	Width of Traffic Lane Marking [m]	Width of Carriageway [m]
Austria RVS 9.232	80-100	3.50	0.15	7.00
Denmark (practice)	90 - 120	3.60	0.10	7.20
France CETU	80-100	3.50	-	7.00
Germany RAS-Q 1996 / RABT 94	100 70 (26t) 110 (29.5T)	3.50 3.50 3.75	0.15 0.15 0.15	7.00 7.00 7.50
Japan Road Structure Ordnance	80-120 60	3.50 3.25	-	7.00 6.50
the Netherlands ROA	120 90	3.50 3.25	0.15 0.15	7.00 6.50
Norway Design Guide Road Tunnels	80-100	3.45	0.10	6.90
Spain Instrucción 3.1	90-120	3.50	0.10	7.00
Sweden Tunnel 99	70 90 110	3.50 3.75 3.75	0.10 or 0.15 0.15 0.15	7.00 7.50 7.50
Switzerland (SN 640201)	80-120	3.50-3.75	0.20	7.75
UK TD27 (DMRB 6.1.2)	110	3.65	0.10	7.30
USA AASHTO	-	3.60	n.s.	7.20

Table 2. Design speed by functional road classification on road design standards of Korea.

Functional Road Classification		Design Speed [km/h]			
		Local Area			Urban
		Flatland	Hill	Mountain	Area
Freeway/Expressway		120	110	100	100
City Street	**Major Arterial**	80	70	60	80
	Minor Arterial	70	60	50	60
	Collector	60	50	40	50
	Local	50	40	40	40

In this section, the design of the adaptive ROI is divided into three subsections: normal driving, lane change driving, and intersection driving. Figure 3.4 shows the process of selecting each ROI in consideration of driving status information, driving environment, and the activation function of the subject vehicle. As defined in Chapter 2, the automated driving level can be determined according to the function and environment applied to the current system. The basic level is always applied as primary mode because the only lane-keeping function is active. In the case of the motorway Level, if the vehicle maintains without changing lanes, the primary mode of setting the detection importance toward the front is applied like the basic level. In the case of performing lane change, the ROI for a lane change in which the front, rear, and lateral ROIs are extended is applied. Urban Level includes all of the functions of the lower levels and applies additionally designed areas when driving at intersections. The previously designed ROI has limitations when driving in an environment where roads intersect or diverge, such as intersections, because the vehicle takes into account general driving conditions (roads with moderate curvature). When driving at the intersection, it is necessary to perceive the front side and the rear side area at the time of entry, and additionally design an intersection ROI to cope with the road crossing situation and activate it in the relevant situation. The process of determining this mode is depicted in flow chart form in Figure 3.4.

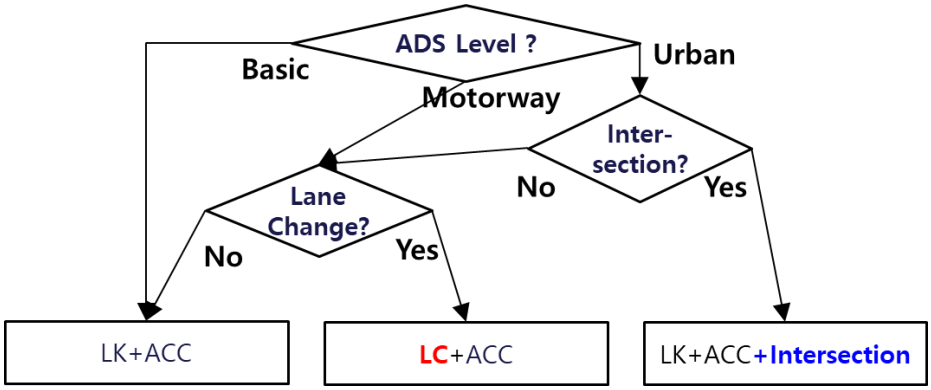


Figure 3.4. Flow chart of driving mode determination according to driving task by automated driving system level.

3.1.1. ROI Design for Normal Driving Condition

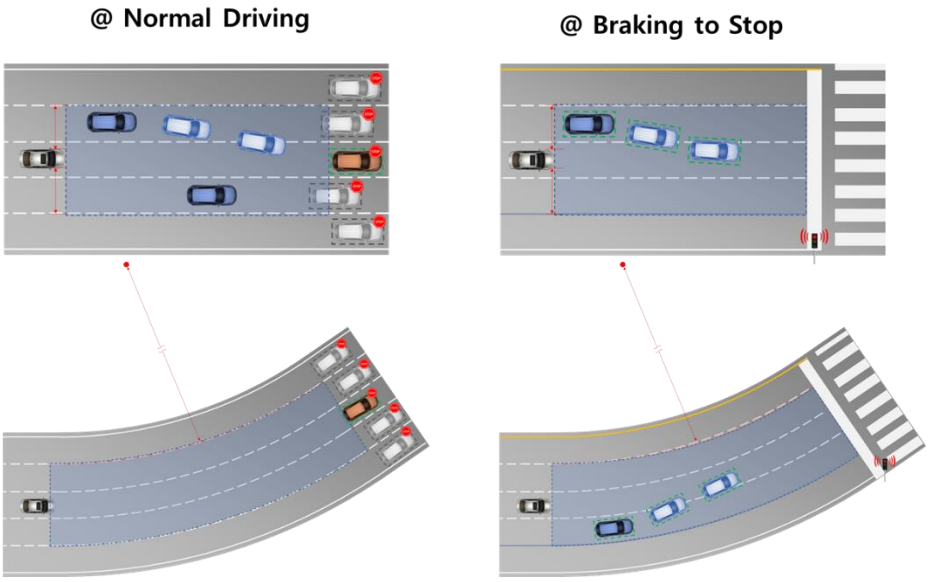
In this subsection, the ROI of the Basic Level is designed to represent the most primary driving situation of the three levels. At this level, autonomous vehicles keep in the lane along the path or lane and avoid collisions with forward obstacles through proper longitudinal control. It can be classified as the most basic and essential driving situation. Reasonable cognitive ROI is designed based on vehicle state, road design criteria (RDS), international standards (ACC: Adaptive Cruise Control, FVCMS: Forward Vehicle Collision Mitigation System), lane information (if reliable), and vehicle driving characteristics studies. Lane information may not be used in an environment where lanes are lost, failure to recognize due to the performance limitation of lane-detecting sensors, or where there are no lanes such as intersections, crosswalks, and parking lots. Thus, for realistic and reasonable design, ROI is designed based on road design standards without lane information, and it is reflected when reliable lane information exists.

To analyze and reflect the driving conditions and characteristics of each area, the ROI design is divided into three area elements, front, rear, and around the vehicle. It is defined as the 1st, 2nd, and 3rd level of ROI according to the importance of each region. Since ROI designed in this subsection is the minimum required perception area in the most normal driving situation, it becomes a basic structure that should be monitored at all times for the normal operation of the automated driving system.

3.1.1.1. *Front ROI Design*

In order to define the area of forwarding interest when driving, two typical situations of driving conditions could be considered. The conditions of normal driving, avoiding collisions with other vehicles and obstacles, and the conditions of braking and stopping due to road facilities, traffic regulations, or obstacles shall be included in the design of the ROI as described in Figure 3.5.

The most important thing from the point of view of securing the driving safety of the vehicle is to avoid collisions with objects along with the path of the vehicle. In general, there are two situations in which objects exist within the path where the vehicle is driven on the road, affecting the behavioral plan. The ACC mode of driving along the preceding vehicle as shown in Figure 3.5(a), and the braking mode of stopping the target object as depicted in Figure 3.5(b), should be considered.



(a) ACC Mode (b) Braking Mode

Figure 3.5. Major representative driving conditions to ensure forward safety and avoid rear-end accident.

To design the required perception area for the front, the radius of curvature of the road according to the longitudinal distance, the lateral distance, and the present speed with respect to the front is determined. Depending on the longitudinal and lateral distances, the forward ROI can be determined in a rectangular shape. Considering the curvature along the curve radius of the road, the region of interest is defined by the concept shown in Figure 3.6. Also, the three typical parameters required for the design of the front ROI are shown in Figure 3.7.

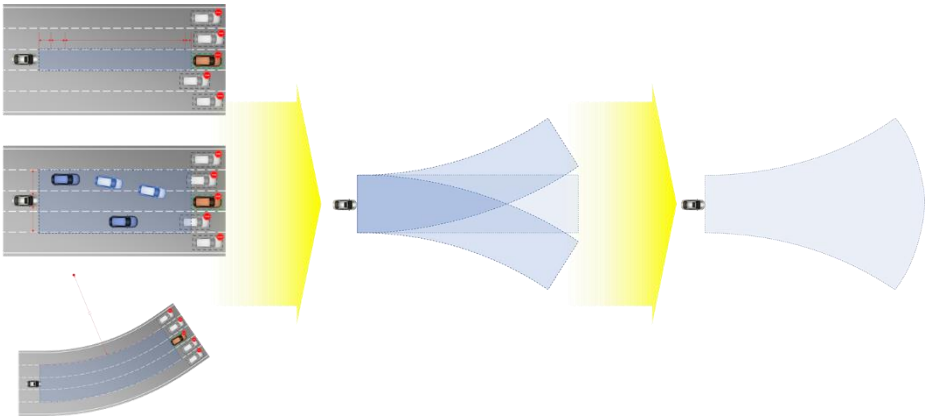


Figure 3.6. Design scheme of front ROI.

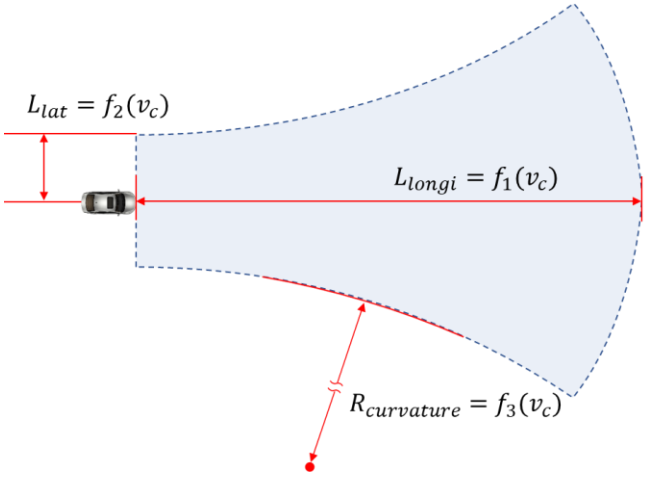


Figure 3.7. Detailed design parameter of front ROI.

Using three parameters in Figure 3.7, the detailed ROI according to importance with or without lane information is defined as shown in Figure 3.8.

If the lane on the front road can be detected through the front vision sensor, the lane area in which the own vehicle is driving can be precisely identified. If lanes are detected, the forward lanes can be specified to focus on a narrower

range than the Road Design Standards. Even if the higher ROI decreased due to lane information, the overall perception ROI is not reduced because the lower ROI acts as a backup zone by covering it. In Figure 3.8, the right lane is detected, and the upper-level area of the right is reduced compared to the left, but the lower-ROI is substituted to ensure that no ROI loss occurs.

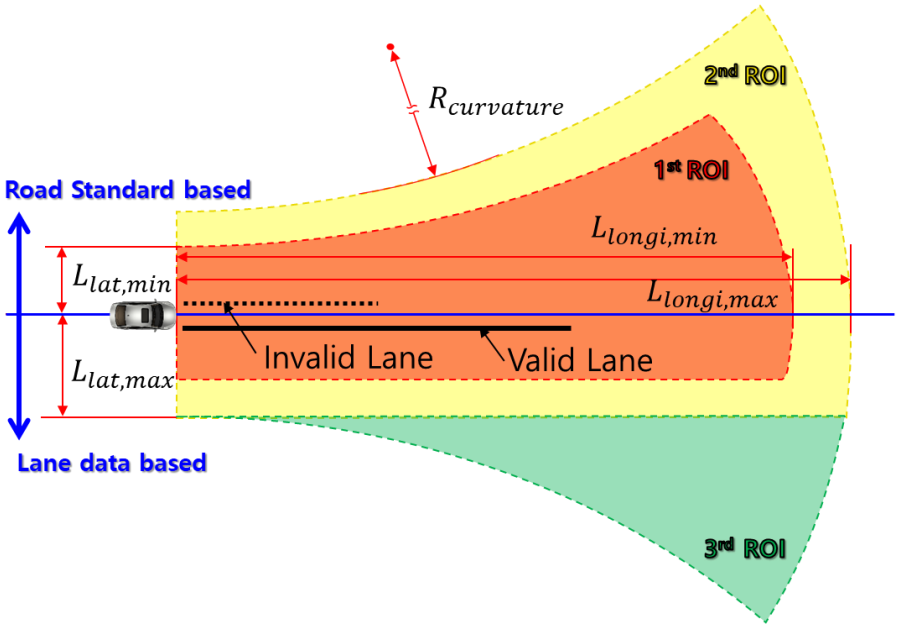


Figure 3.8. Designed ROI by important level under reliable right lane detection.

When braking by a forward object, the braking distance is shorter with the AEB because the braking command is higher in the AEB than in the ACC. However, in this study, the longest braking distance is reflected in the calculation of the forward minimum recognition distance because the aim is to secure the minimum recognition area to ensure safety. In the case of ACC mode,

the limit of acceleration and jerk $(2.5\mathrm{m}/s^3)$ is limited in ISO-ACC. To cope with the extreme ACC situation to avoid accidents, the longitudinal safety distance that can be secured by considering various factors under the ACC mode condition when the vehicle ahead is suddenly stopped is calculated as follows.

$$L_{longi,\,front,Braking} = v_c \cdot t_{detect} + v_c \cdot t_{delay} - \frac{1}{6} jerk \cdot t_{delay}^3 + \frac{v_c^2}{2a_{decel,ACC}} + c_{front,min} \quad (3.1)$$

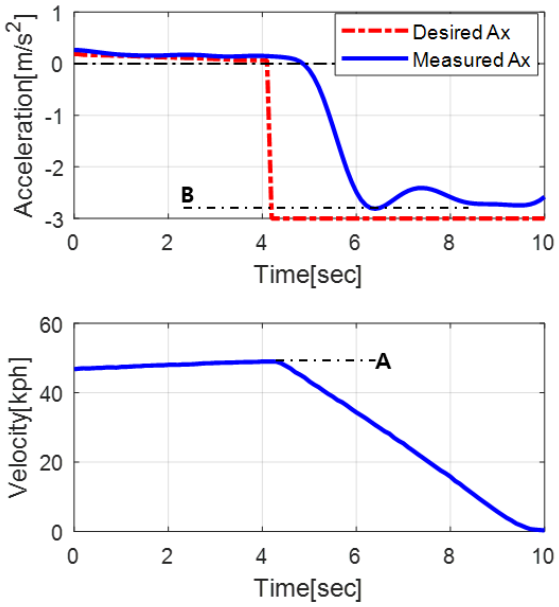
where, v_c is current velocity of subject vehicle and t_{delay} is the actuator delay time of the subject vehicle. $c_{front,min}$ means minimum clearance to the front, and t_{detect} is the required minimum detection time of object detection algorithm. $a_{decel,ACC}$ is maximum deceleration limit of ACC function.

Besides, the recommended time gap, TG_{safe}, for vehicle driving is usually 2 seconds. For safety purposes, a wider range of longitudinal cognitive distances is calculated by reflecting more as much as this time gap in $L_{longi,\,front,Braking}$ of Equation (3.1).

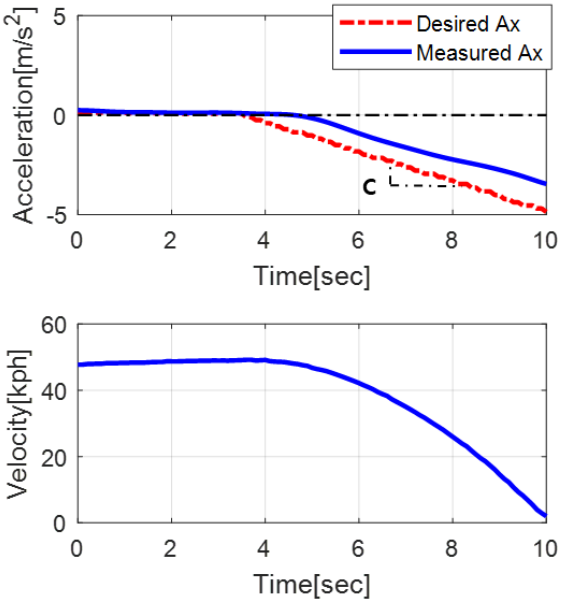
$$L_{longi,\,front,Safe} = v_c \cdot \left(t_{detect} + TG_{safe}\right) + v_c \cdot t_{delay} - \frac{1}{6} jerk \cdot t_{delay}^3 + \frac{v_c^2}{2a_{decel,ACC}} + c_{front,min} \quad (3.2)$$

The t_{delay} means the actuator delay time of the subject vehicle and takes up a great deal of weight in calculating the braking distance. By performing system identification on the longitudinal control characteristics of the test vehicle, the actual delay time can be calculated. Although actuator delay time analysis is

required for both acceleration and deceleration situations, only deceleration was considered in this case as it is necessary to calculate the clear braking distance for collision prevention. In the deceleration situation, system identification is performed by acquiring the actual measured longitudinal acceleration data according to the deceleration command of the actuator through actual experiments. The results are calculated for two types of inputs, referring to the use of step and ramp inputs in general when performing system identification, as shown in Figure 3.9. In order to fit the model for the step input, the data was obtained by adjusting the final speed before the start of deceleration input, A in Figure 3.9(a), and the target acceleration after inputting the deceleration command, B in Figure 3.9(a). The slope C in Figure 3.9(b) with respect to the target acceleration was variously set for the ramp input.



(a) Step input (deceleration)



(b) Ramp input (deceleration)

Figure 3.9. Example driving log of acceleration and velocity of test vehicle.

Since we do not know the actual longitudinal control system model of the vehicle, we perform the system identification on three models: Simple time delay model (STD), first-order plus time delay model (FOPTD), and second-order plus time delay model (SOPTD). The System Identification Toolbox of the MATLAB was used to fit the data log into each model, and as a result the FOPTD model was found to match the most. The equation for the FOPTD model is:

$$\text{FOPTD:}\quad \frac{Y(s)}{U(s)} = \frac{Ke^{-T_d s}}{1+T_p s} \tag{3.3}$$

where T_p is process time constant, T_d is process time delay, and K is process gain set to 1 in this study.

The model fitting results for each input are shown in Table 3, Table 4, and Table 5.

Table 3. System identification result of longitudinal control by applying FOPTD model with step input at 10kph driving.

Speed before deceleration [km/h]	Desired deceleration $[\mathrm{m}/s^2]$	Process time constant, T_p [s]	Min. actual deceleration $[\mathrm{m}/s^2]$	Process time delay, T_d [s]
10	-1	0.5	-0.96	1.0
	-2	0.6	-1.8	
	-3	0.7	-2.2	
	-4	-	-2.4	
	-5	1.0	-2.4	
	-6	1.7	-2.4	

Table 4. System identification result of longitudinal control by applying FOPTD model with step input at 30kph driving.

Speed before deceleration [km/h]	Desired deceleration $[\mathrm{m}/s^2]$	Process time constant, T_p [s]	Min. actual deceleration $[\mathrm{m}/s^2]$	Process time delay, T_d [s]
30	-1	0.5	-0.95	1.0
	-2	0.6	-1.9	
	-3	0.6	-2.9	
	-4	0.6	-3.6	
	-5	0.8	-4.4	
	-6	1.1	-4.4	

Table 5. System identification result of longitudinal control by applying FOPTD model with step input at 50kph driving.

Speed before deceleration [km/h]	Desired deceleration $[\mathrm{m}/s^2]$	Process time constant, T_p [s]	Min. actual deceleration $[\mathrm{m}/s^2]$	Process time delay, T_d [s]
50	-1	0.5	-0.95	1.0
	-2	0.5	-1.9	
	-3	0.5	-2.8	
	-4	0.7	-3.5	
	-5	0.8	-4.2	
	-6	1.1	-4.6	

Table 6. System identification result of longitudinal control by applying FOPTD model with ramp input.

Slope of deceleration $[\mathrm{m}/s^3]$	Process time constant, T_p [s]	Min. actual deceleration $[\mathrm{m}/s^2]$	Process time delay, T_d [s]
-0.25	0.9	-2.2	0.7
-0.5		-3.0	
-0.75		-3.6	
-1.0		-3.9	

From the system identification result above, the time delay of the test vehicle was determined to be 1.0 sec and 0.7 sec for the step input and ramp input, respectively. Since this analysis is used to determine the braking distance, a larger value of 1.0 seconds is used to design ROI.

Harwood suggested that a sufficient Acceptance Time Gap, TG_{turn}, needed for a vehicle to turn left or right at an intersection, is 7.5 seconds [Harwood,'99].

It may be determined as a driving safety distance according to the speed of the ego vehicle in a road environment in which the other vehicle is not in a straight line where interference with the progress path of the own vehicle may occur. Therefore, we can define $L_{longi,\,front,Turn}$ as:

$$L_{longi,\,front,Turn} = v_c \cdot TG_{turn} \tag{3.4}$$

Since these forward ranges are dependent on the speed of the subject vehicle, the driving speed is reduced, and the value becomes considerably smaller when it is close to a standstill. The minimum safety distance for the overtaking vehicle is determined by referring to the statistical data of the driving data coming in front of the overtaking vehicle and the lower limit of $L_{longi,\,front,\min}$.

$$L_{OT,after\,headway} = m_{OT,after\,headway} + 2 \cdot \sigma_{OT,after\,headway} \tag{3.5}$$

Using this defined longitudinal safety distance, $L_{longi,\,front,\min}$ and $L_{longi,\,front,\max}$ are defined as follows.

$$L_{longi,\,front,\min} = \max(L_{longi,\,front,Extend}, L_{longi,\,front,Turn}) \tag{3.6}$$

$$L_{longi,\,front,\max} = \max(L_{OT,after\,headway}, L_{longi,\,front,Braking}) \tag{3.7}$$

The determination of parameters for designing the transverse safety range is made as follows. In the case of the basic level, lane-keeping is carried out, and

deceleration and acceleration control are performed based on the relative speed and the relative distance to the object in the own lane. It is necessary to monitor strictly for safety since the vehicle can be driven into the lane from the adjacent lane on either side of the lane. The $L_{lat,min}$, therefore, means the width from the lane center to the lane outside the adjacent lane. The Lane width is basically determined by referring to lane width limit value by the design speed of Road Design Standards in Table 7. If valid lane information is detected by the front vision, the obtained lane information is used.

Table 7. Minimum road width by road classification and design speed in road design standards of Korea.

Road Classification			**Minimum Road Width [m]**		
			Local Area	Urban Area	Compact Car Road
Freeway/Expressway			3.50	3.50	3.25
City Street	**Design Speed [km/h]**	**≥ 80**	3.50	3.25	3.25
		≥ 70	3.25	3.25	3.00
		≥ 60	3.25	3.00	3.00
		< 60	3.00	3.00	3.00

The lane width based on RDS, $w_{RDS\,lane}$, and the minimum lateral range, $L_{lat,min}$, are determined as follows.

$$w_{RDS\,lane} = \begin{cases} 3.50 & (v_c \geq 80) \\ 3.25 & (60 \leq v_c < 80) \\ 3.00 & (v_c < 60) \end{cases} \tag{3.8}$$

$$L_{lat,min} = \begin{cases} \frac{3}{2} \cdot w_{[i],vision\,lane} & (lane_{confid[i]} > 1) \\ \frac{3}{2} \cdot w_{[i],RDS\,lane}(v_c) \end{cases} \tag{3.9}$$

$$where,\ i = \{Left, Right\}$$

where $lane_{confid[i]}$ represents the confidence level of lane obtained from front vision which means reliability of lane detection. It has a integer value from 0 to 3, and the larger the number, the more reliable.

Occasionally, there are vehicles that drive outside of both sides of the lane that cross more than one lane, so it is possible to set up $L_{lat,max}$ that extends by the width of the lane to monitor it.

$$L_{lat,max} = \begin{cases} \frac{5}{2} \cdot w_{[i],vision\,lane} & (lane_{confid[i]} > 1) \\ \frac{5}{2} \cdot w_{[i],RDS\,lane}(v_c) \end{cases} \tag{3.10}$$

$$where,\ i = \{Left, Right\}$$

In most well-organized cities, roads are designed in a straight line, but in general, roads are usually curved. Road curvature can be calculated with the approach of RDS and ISO-ACC, respectively. In road design criteria, the curvature of a road is determined by the driving design speed of the road. The automated vehicle can estimate the maximum radius of curvature of the

currently running road under the premise that the driving control is optimized for road design. According to the Table 8, the minimum plane curve radius, $r_{curvature,RDS}$, according to the maximum slope can be obtained through interpolation by the design speed of the road. The maximum slope of the road was assumed to be 8% in this study.

Table 8. Minimum plane curve radius per design speed of road design standard of Korea.

Design Speed [km/h]	**Minimum Radius Curvature [m]**		
	Maximum Superelevation Slope		
	6%	7%	8%
120	710	670	630
110	600	560	530
100	460	440	420
90	380	360	340
80	280	265	250
70	200	190	180
60	140	135	130
50	90	85	80
40	60	55	50
30	30	30	30
20	15	15	15

In addition, ISO-ACC restricts lateral acceleration of vehicles to $3\mathrm{m}/s^2$. Velocity according to lateral acceleration and curvature limitation is presented as follows.

$$v_{circle} = \sqrt{a_{lateral\_curve} \cdot R_{\min}} \tag{3.11}$$

where, v_{circle} is a steady-state speed for the curve driving, $R_{\min}$ is curve radius, and $a_{lateral\_curve}$ is the design lateral acceleration for curves on highways which is derived from average driver behavior in curves (95% drivers) in Mitschke ('91).

Using the relationship between the vehicle speed and the maximum lateral acceleration, the road curve radius is derived as follows.

$$r_{curvature,ISO-ACC}(v_c) = \frac{v_c^2}{a_{lateral,curve}} \tag{3.12}$$

The smaller the radius of curvature of the road, the greater the curvature, which can be considered to be the wider the driving range. Therefore, by selecting the smaller value of the two curvature radii of the roads thus obtained, the road area with greater curvature is included as the region of interest.

$$r_{curvature} = \max\left(r_{curvature,RDS}(v_c) \quad r_{curvature,ISO-ACC}(v_c)\right) \tag{3.13}$$

As described above, we designed an anterior region of interest at the 1st, 2nd, and 3rd level using RDS. It is a method of estimating the range of roads that can be driven, provided that the vehicle is on the road. When defining the lateral range in Equation (3.9), it also applies to road curvature along the lane, as defined by a decent lane detection level.

3.1.1.2. *Rear ROI Design*

The minimum perception range at the rear of the vehicle is designed in detail. In the case of basic level, it is not necessary to consider driving situations such as lane change and turn, so it is necessary to consider the situation in which own vehicle is a threat during lane-keeping driving. Consideration should be given to a lane change from the rear to a lane ahead of the ego vehicle. In the ADS of this study, the minimum recognition time is necessary to build up the tracking reliability for reliable recognition of the surrounding objects. Even if there is no lane change or turn motion, the minimum rear detection should be performed because the safety of the ego vehicle can be secured by monitoring the approaching vehicle from the rear side. Therefore, the rear ROI in normal driving conditions is designed, taking into account the characteristics of overtaking driving behavior. As with the design of the front ROI, the importance defines the ROI by dividing it into 1st level and 2nd level.

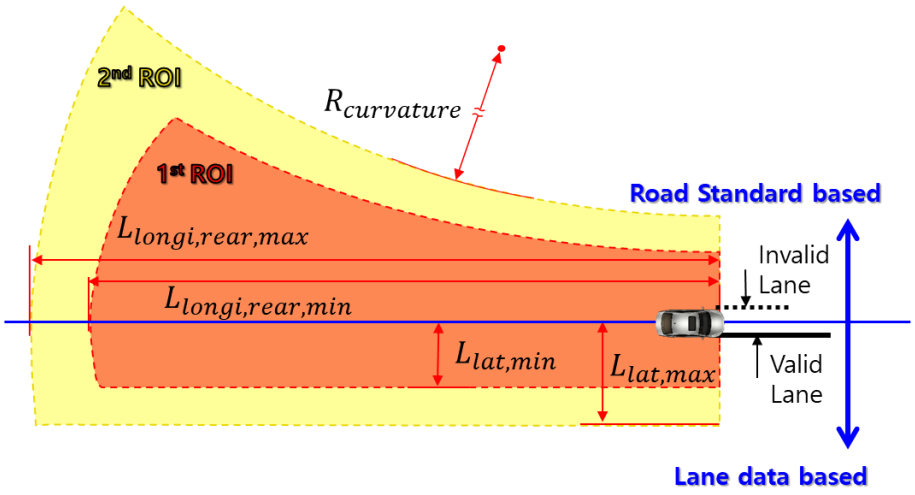


Figure 3.10. Detailed design parameter of rear ROI.

In the case of the rear ROI, the lateral range and the radius of curvature of the road are calculated in the same way as the front, using the state variables of the ego vehicle, as described in Figure 3.10. Since the radius of the curve is determined by the speed of the vehicle, the same $r_{curvature}$ calculated earlier applies. The same applies to $L_{lat,min}$ and $L_{lat,\max}$ in (3.9) and (3.10) calculated from the lateral width of front ROI design.

As mentioned earlier, monitoring of vehicles attempting to overtake from the rear is necessary, so this section considers how far perceptible the rear is in the longitudinal direction. The proper distances necessary for determining the minimum detection range of several rear longitudinal distances are calculated, of which an appropriate value is selected.

Hegeman ('04) analyzed the maneuver characteristics of the overtaking vehicle through the majority of driving data of the over-taking vehicle. Overtaking strategies were performed based on the findings of Gordon ('70), who classified the characteristics of overtaking vehicles. Since the distance prior to the overtaking maneuver distribution for forward passing vehicles is a normal distribution, the 2-sigma range is used.

$$L_{prior\,headway} = m_{prior\,headway,o} + 2 \cdot \sigma_{prior\,headway,o} \tag{3.14}$$

We used the prior headway among the time gap of prior headway for each strategy.

$$L_{prior\,TG\,headway} = v_c \cdot \max\left(m_{TG,\,flying} \quad m_{TG,\,piggi\,backing} \quad m_{TG,2+}\right) \tag{3.15}$$

where, $m_{TG,\,flying} = 2.503\,(p < 0.05)$, $m_{TG,\,piggi\,backing} = 2.805\,(p < 0.05)$, $m_{TG,2+} = 2.494\,(p < 0.05)$.

Comparing the two values calculated so far, the $L_{longi,rear,min}$ and $L_{longi,rear,\max}$ required for the rear 1st Level ROI can be defined as follows.

$$L_{longi,rear,min} = \min\left(L_{prior\,headway} \quad L_{prior\,TG\,headway}\left(v_c\right)\right) \tag{3.16}$$

$$L_{longi,rear,\max} = \max\left(L_{prior\,headway} \quad L_{prior\,TG\,headway}\left(v_c\right)\right) \tag{3.17}$$

3.1.1.3. *Surrounding ROI Design*

So far, we have defined areas of interest for the front and rear of the vehicle. We decide the area to monitor the surrounding area that should be minimally recognized regardless of the vehicle status.

The front and rear areas of interest defined so far are approaches for recognizing nearby objects or facilities according to the direction of the vehicle. This approach is appropriate for motorways, such as highways and arterial roads. However, in the urban environment, various objects, such as pedestrians and two-wheelers, exist in addition to vehicles and facilities, and an unexpected sudden situation may occur under various environmental conditions, and an expanded area of interest is required. Therefore, the third-level area of interest defines the surround ROI that always performs recognition regardless of the state of the vehicle. Since the driving direction of the vehicle is the front and

rear directions, an elliptical ROI having a major axis in the forward direction is proposed, as depicted in Figure 3.11.

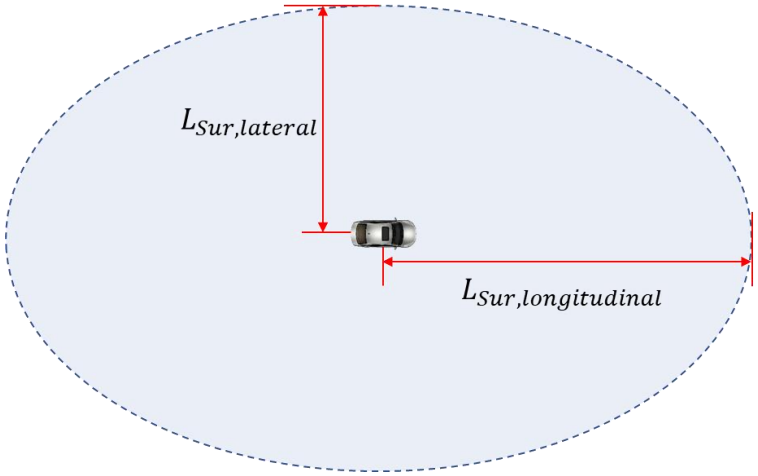


Figure 3.11. Detailed design parameter of ellipsoid based surrounding ROI.

In equation (3.5), the result of calculating the minimum distance that should be secured in the longitudinal direction is set to the length of the long axis of the ellipse to be designed.

$$L_{Sur,lat} = m_{OT,after\ headway} + 2 \cdot \sigma_{OT,after\ headway} \tag{3.18}$$

It also considered the possibility of a running pedestrian appearing to determine the lateral range. In urban environments, there is a possibility that pedestrians may appear running from various directions, such as jaywalking, so we want to ensure sufficient monitoring of this situation. A statistical analysis of the place state according to the pedestrian velocity distribution shows that

the average speed of a running pedestrian is 6 m/s [Tordeux,'16, Oh,'19]. It is assumed that the Object tracking algorithm had a sufficient time of 4 seconds to recognize pedestrians and properly predict their behavior. Based on this information, we can calculate the minimum distance of the area to be monitored at all times and use it as the minor axis length of the surrounding ellipse.

$$L_{Sur,longi} = v_{ped,running} \cdot t_{suff,detection} \tag{3.19}$$

In Figure 3.12, the results of integrating the front, rear and surrounding areas of interest so far designed by importance can be seen. The speed adaptive ROI is designed when there is left lane information in general lane keeping driving.

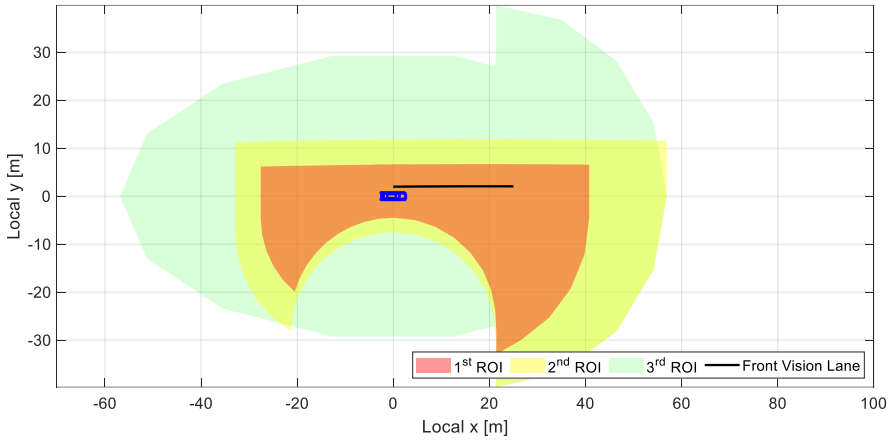


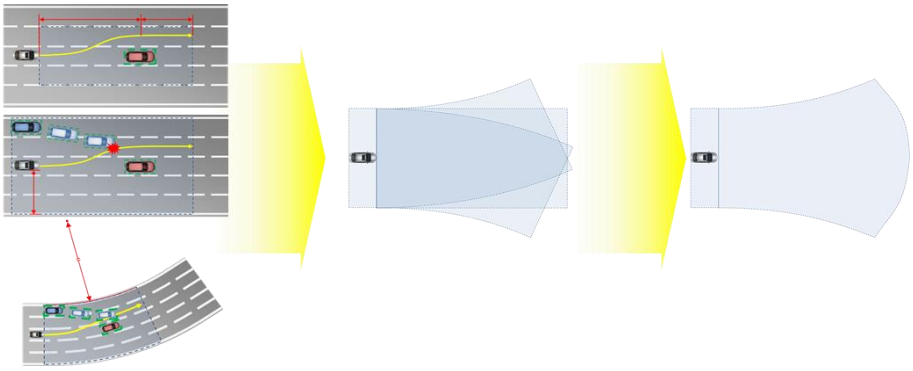
Figure 3.12. ROI integration scene for normal driving with reliable left lane.

3.1.2. ROI Design for Lane Change

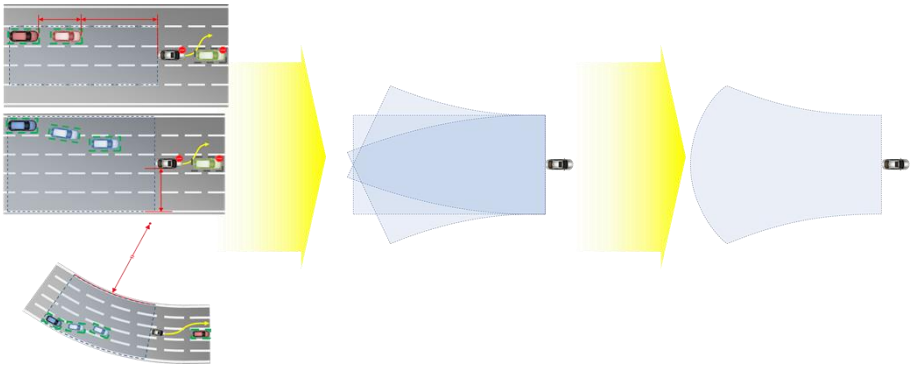
Previously, the ROI was designed in the basic driving mode, which maintains lanes and prevents front collisions. However, since not all roads are single-lane, lane changes are essential for safe and competent driving to the destination. In the case of Tesla's Autopilot currently on the road, if the driver commands the lane change signal through the direction indication lever, it checks whether the lane change behavior is safe and performs the lane change. Automated driving studies have been conducted to determine whether to change lanes in various situations and to perform lane change effectively. Suh proposed how to determine and perform a lane change decision when an autonomous vehicle needs to change lanes for efficient driving or to change lanes at a merger or branch road [Suh,'16]. If the automated driving system aims to change lanes in a specific direction, it is possible to determine if the lane change is possible and design an expanded area of interest to secure safety. Although similar to how ROI was designed for front and rear in normal driving situations in 3.1.1.1 and 3.1.1.2, the areas of interest are extended to reflect additional considerations when changing lanes.

A safe lane change requires avoiding collisions between vehicles in front lanes and vehicles in destination lanes. In normal driving, since the ego lane is the purpose lane for driving, the ROI is set as the area in which the vehicles in the ego lane or the potential for entering the ego lane exist. In the case of a lane change, the target lane is adjacent, not an ego lane, so it is essential to monitor the vehicles existing in the target lane and potentially entering vehicles. Figure 3.13 shows the extended ROI design scheme in the front and rear of lane-change

driving. Figure 3.13 (a) defines the area of interest for changing lanes in the forward lane by reflecting the vehicle in the forward lane, the vehicle potentially entering the target lane, and the curvature of the road. Similarly, Figure 3.13 (b) shows the definition of a region of interest, taking into account the vehicle driving on the target lane, the vehicle potentially entering the target lane and the curvature of the road. Thus, the transverse required perception range extends one lane wider than the basic ROI. Both sides are expanded in Figure 3.13, but in a situation where the lane change direction is determined, the area is expanded only in the direction for efficiency. Figure 3.14 shows that the ROI expands as you change lanes from normal driving to lane change when you try to change lanes to the right. Also, the parameters used to determine ROI in the situation are shown.

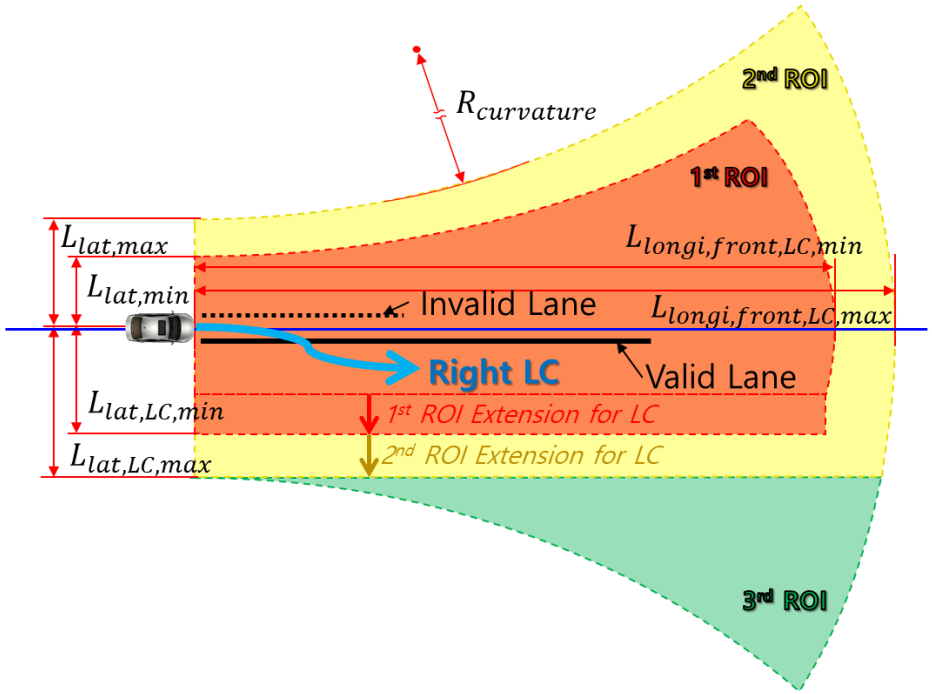


(a) Front ROI for lane change driving

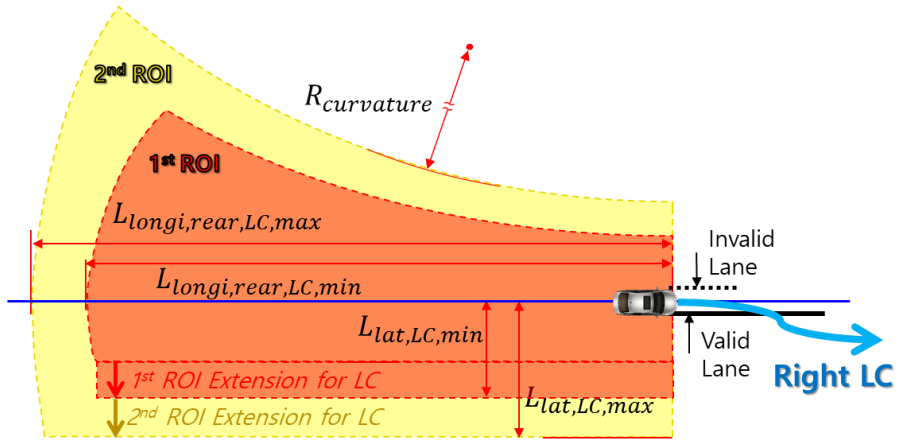


(b) Rear ROI for lane change driving

Figure 3.13. Designed ROI for lane change driving.



(a) Front ROI



(b) Rear ROI

Figure 3.14. Detailed design parameter for lane change driving.

First, when changing lanes, it is necessary to determine the recognition range of the target lane to calculate the required recognition range in the forward longitudinal direction. Based on the statistical analysis of lane-change driving data, the lane change distance and the spacing of the vehicle in front of the vehicle are determined by using the time required for lane change and the spacing characteristics of the vehicle in the front and side lanes [Toledo,'07].

$$
\begin{aligned}
&L_{longi,\,frontLC,duration} = v_c \cdot t_{LCduration,total} + c_{LC} \\
&L_{longi,\,frontLC,spacing} = L_{FVspacing} + c_{LC} \\
&where,\ t_{LCduration,total} = m_{LCduration,total} + 1 \cdot \sigma_{LCduration,total} \\
&\qquad L_{FVspacing} = m_{FVspacing} + 1 \cdot \sigma_{FVspacing}
\end{aligned}
\tag{3.20}
$$

By comparing the distance thus calculated and the forward braking distances calculated in equations (3.6) and (3.7), the forward longitudinal distance required for lane change can be determined as follows.

$$
\begin{aligned}
L_{longi,\,frontLC,max} &= \max\left(L_{longi,\,front,max},\ \max\left(L_{longi,\,frontLC,duration},\ L_{longi,\,frontLC,spacing}\right)\right) \\
L_{longi,\,frontLC,min} &= \max\left(L_{longi,\,front,min},\ \min\left(L_{longi,\,frontLC,duration},\ L_{longi,\,frontLC,spacing}\right)\right)
\end{aligned}
\tag{3.21}
$$

The rear longitudinal distance is considered for lag-read spacing to be obtained from the target lane during lane change. In addition, the distance is calculated to reflect the time the vehicle is changing lanes, assuming that the target lane vehicle is travelling at a slightly faster speed than the vehicle [Toledo,'07].

$$
\begin{aligned}
L_{longi,rearLC,duration} &= v_c \cdot t_{LCduration,total} \\
t_{LCduration,total} &= m_{LCduration,total} + 1 \cdot \sigma_{LCduration,total}
\end{aligned}
\tag{3.22}
$$

The maximum value is used for the 1st ROI level by comparing the necessary distances to the rear calculated above. The 2nd ROI rear longitudinal range is decided to expand by 20% to ensure a greater recognition range for safety.

$$
\begin{aligned}
L_{longi,rearLC,min} &= \max\left(L_{longi,rear,\min dist} \quad L_{longi,rearLC,duration} \quad L_{longi,rearLC,Lag-Lead\ spacing}\right) \\
L_{longi,rearLC,max} &= 1.2 \cdot L_{longi,rearLC,min}
\end{aligned}
\tag{3.23}
$$

where, $L_{longi,rearLC,Lag-Leadspacing}$ is the Lag-lead spacing and $L_{longi,rear,\min dist}$ is minimum clearance set to 20m in this study.

In the case of the transverse direction, as mentioned above, the transverse range extended by one lane width than the normal driving situation is determined in order to monitor a vehicle that may potentially enter the target lane for lane change.

$$
L_{[i],lat,LC,max} = \begin{cases} \frac{7}{2} \cdot w_{[i],lane}\left(v_c\right) & (if\ i = LC\ direction) \\ \frac{5}{2} \cdot w_{[i],lane}\left(v_c\right) & (else) \end{cases}
\tag{3.24}
$$

where, $i = \{Left, Right\}$

The road curvature uses the result of Equation (3.13) calculated according to the speed of the vehicle in normal driving.

3.1.3. ROI Design for Intersection

In this subsection, the minimum cognitive requirements in the intersection environment are designed in detail. Although the intersection environment can identify the shape of the intersection when using a map, this study proposes a method to cope with the situation without a map. The intersection environment is also designed to be safe enough within the range of urban driving speed according to the road design standard so that the basic ROI area defined in the above-mentioned normal driving is possible to some extent. At the intersection, however, there are situations in which the direction of travel of the own vehicle is different, and the angles close to 90 degrees should be considered. To safely enter and exit an intersection, an area of interest for further intersection driving before and after the intersection area is required. As shown in Figure 3.15, the ROI is designed to drive the intersection using the intersection angle, width, and length. Finally, Figure 3.16 shows that the ROI is additionally designated to the front and rear diagonal directions to the previously designed normal driving ROI.

Although the size and shape of the region of interest may change depending on the situation, the integrated ROI shape is close to a circle based on the center of the own vehicle. It shows that omnidirectional cognitive performance is essential to ensure safety in complex environments, such as intersections.

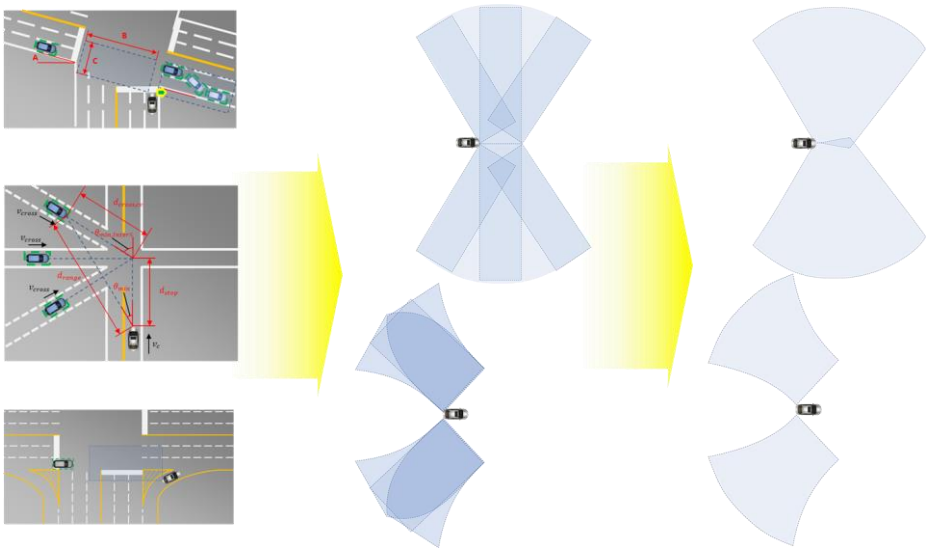


Figure 3.15. Designed ROI at intersection.

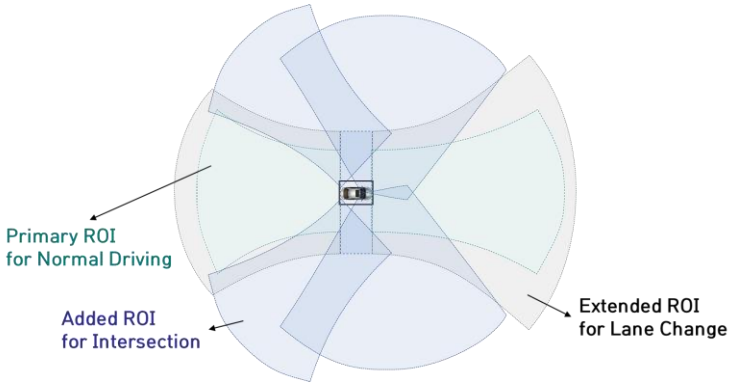


Figure 3.16. ROI integration for intersection driving.

The design parameters of the ROI, which are additionally defined for intersection mapping in Figure 3.15 and Figure 3.16, can be seen in Figure 3.17. Figure 3.17(a) is the front diagonal direction of interest, and the unique point is designed using the maximum minimum angle to recognize the cars on the intersecting road. Figure 3.17(b) shows the ROI of the rear-side to counteract exit from the intersection.

Although the intersections where the compartments are well organized are installed at an angle of close to 90 degrees, many intersections actually have various intersecting angles, so countermeasures are required for these environments. The intersection is designed with a minimum angle of 60 degrees between the road entering and considering the driving ability and safety of the vehicle[Gattis,'98]. As shown in Figure 3.18, it is assumed that the intersection angle is located between 60 and 150 degrees and that the vehicle entering the intersection runs at a speed of about 20% higher than the own vehicle for safety. It is possible to calculate the distance required for emergency braking of the own vehicle, d_{stop}, and the travel distance of another vehicle approaching the intersection, $d_{cross,cv}$, during the time required for braking is traveling at 20% speed. Use this to calculate that collisions do not occur at the point where the vehicle is expected to be a potential crossing point.

$$\begin{aligned} d_{stop} &= \frac{v_c^2}{2a_{decel}} \\ d_{cross,cv} &= v_{cross}\left(t_{stop} + t_{perc,req}\right) + l_v \end{aligned} \tag{3.25}$$

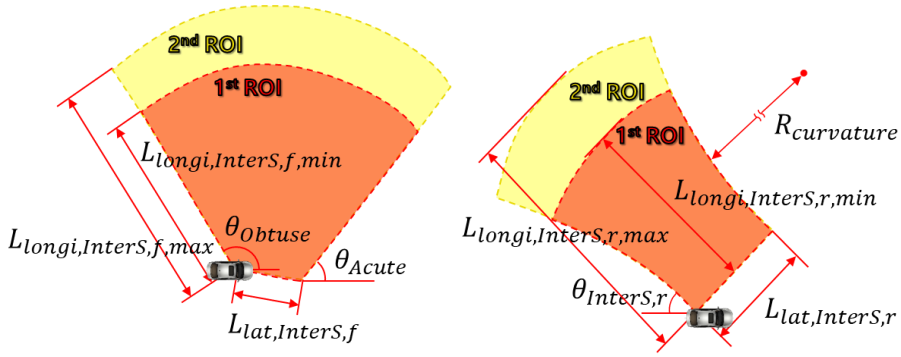
where v_{cross} is speed of vehicle entering intersection from a far side of minor road which is set to 120% of subject vehicle speed. l_v is the vehicle length set to 5 meters.

The angle to detect the furthermost vehicle approaching the intersection from ego vehicle, $\theta_{\min}$, can be obtained using the geometry of the triangle. In addition, a second cosine formula can be used to calculate the d_{range}.

$$\theta_{\min} = \tan^{-1}\left(\frac{d_{cross,cv}\cdot\sin\left(\theta_{\min,InterS}\right)}{d_{stop}+d_{cross,cv}\cdot\cos\left(\theta_{\min,InterS}\right)}\right) \tag{3.26}$$

$$d_{range} = \sqrt{{d_{stop}}^2 + {d_{cross,cv}}^2 - 2d_{stop}d_{cross,cv}\cdot\cos\left(\pi-\theta_{\min,InterS}\right)} \tag{3.27}$$

where $\theta_{\min,InterS}$ is minimum intersection angle in road design standards, which is set to 60 degrees in this research.



(a) Forward ROI for intersection (b) Rear ROI for intersection

Figure 3.17. Detailed design parameters for intersection ROI.

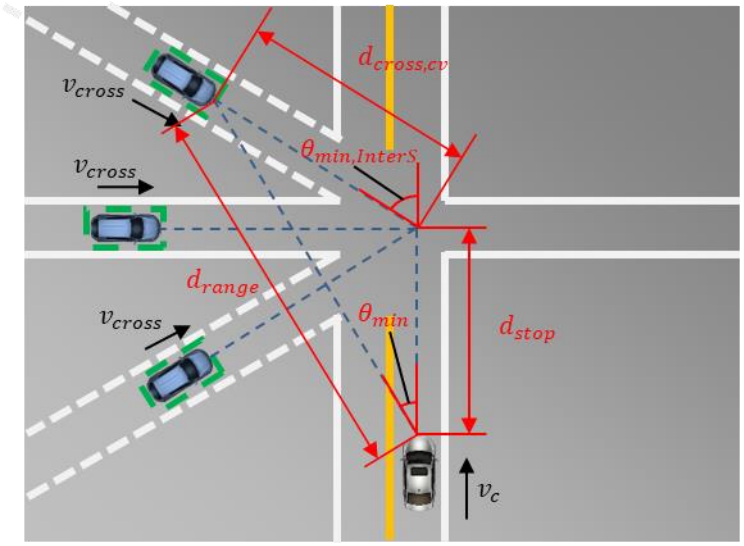


Figure 3.18. Scheme of angle and distance calculation under collision assumption at the intersection.

Using Equations (3.26) and (3.27), define the parameters needed to build an intersection ROI, as shown in Figure 3.17.

In the forward direction, the width of the intersection road is defined as three times the lane width, and the minimum longitudinal range uses a value of (3.27). For safety, the extended 2nd level longitudinal range is designed to be 150% of the minimum longitudinal range. θ_{acute} is applied to $\theta_{\min}$ obtained in (3.26), and θ_{obtuse} is fixed at 120 degrees, considering that the design angle of the intersection was at least 60 degrees, as previously noted.

In the case of the rear direction, it is similar to the method of defining the front and rear ROI defined above. The ROI is defined by reflecting the longitudinal range, lateral range and curvature of the road. The difference is to

secure the rear region of interest by reflecting the angle to detect the vehicle in the relative lane crossed while driving at the intersection. In this study, the angle is set to 35 degrees.

It is inefficient to activate regularly the intersection ROI designed above. Since the road design dictates that the diamond road marker is displayed before the intersection of the road, this marker recognition gives a reasonable prediction of the forward intersection situation. In this way, it is expected that safe and efficient ROI selection will be possible by extending the intersection response ROI in the situation where the intersection driving situation can be predicted in various ways such as V2X and traffic light recognition in the future.

Figure 3.19. The diamond-shaped road marker indicating ahead intersection.

3.2. Data Processing based on Adaptive ROI

The batch processing by applying the same weight to all surrounding information causes a large computational load on the entire system. In order to solve this problem, we propose a method to reduce the computational load without degrading the driving safety of the vehicle by applying the weight of each region designed in the previous Section 3.1.

In the previous section, we designed an adaptive ROI by importance to ensure driving safety based on vehicle speed and detected lane information. This section deals with how to process environment information obtained from the lidar sensor using the constructed adaptive region of interest. Lidar information is acquired in the form of the point cloud and processed into a form that represents the surrounding environment necessary for motion planning. The processing of point cloud consists of voxelization, which typically performs data downsizing, and clustering, which classifies point group signals by object. In order to achieve more efficient and advanced performance, this section categorizes point cloud by Adaptive ROI prior to these processes and proposes processing methods that reflect the important characteristics of each region by using this result. The designed adaptive ROI-based data processing scheme is shown in Figure 3.20.

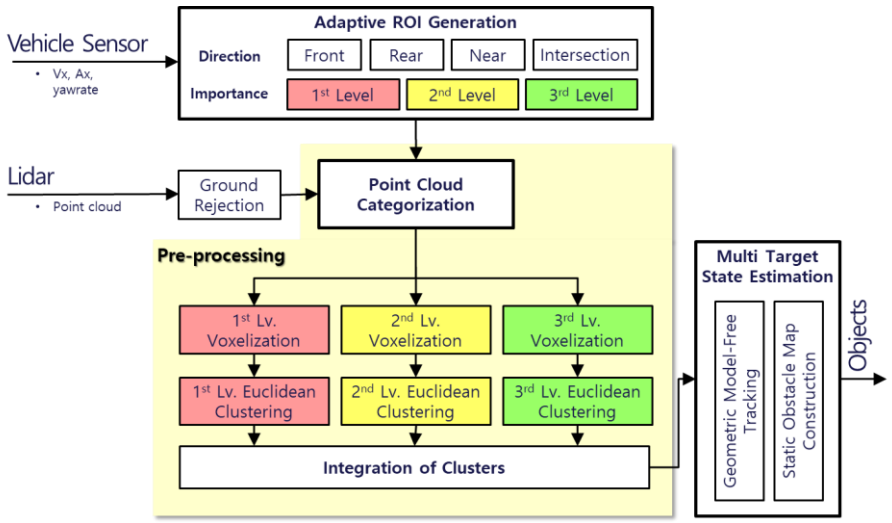


Figure 3.20. Scheme of point cloud processing based on adaptive ROI.

3.2.1. Point Cloud Categorization by Adaptive ROI

Recently, many automated vehicles utilize precision road maps constructed to process a large amount of lidar information. Using road areas such as roads and sidewalks on the map, the point cloud is classified by setting ROI only in areas where objects such as people or vehicles can exist. This approach is highly efficient on the premise that map information represents the actual road environment, but it can pose a significant risk if the degraded map information in the vehicle localization differs from reality, for example, due to climate, satellite conditions, and so forth. It is not applicable in areas where maps are not constructed, and if an unexpected situation occurs or the actual road repair or change is not updated on the map, improper ROI setting may cause problems with the performance of environment perception.

We classify point cloud acquired from the lidar sensor using the previously designed adaptive ROI. Since the shape and size of the region of interest vary depending on the driving condition of the vehicle and the reliability of the lane recognition, area-specific point cloud classification shall be carried out for each cycle. The goal is to achieve perception performance through efficient pre- and post-processing execution strategies because it is limited to performing all point cloud processing under limited resources. To secure the driving safety of automated driving vehicles by ensuring sufficient precision and accuracy without omission or distortion of the information recognized by the sensors, the process of classifying the point cloud by the designed adaptive ROI must precede.

In most cases, areas of high importance are included in areas of low importance. Therefore, point cloud categorization is performed in the order of importance of the region of interest to increase computational efficiency. If the points exist within the region of interest, categorization is applied in such a way that an importance index is assigned to each point. In this study, the point cloud without the ROI importance index is considered as not necessary right away from the viewpoint of driving safety and is eliminated. In the future, if applied to hardware with higher cognitive module computational capability, the areas of perceived importance can be expanded by considering the eliminated points as the fourth area of importance. The results of point cloud categorization by ROI level are shown in Figure 3.21, Figure 3.22, and Figure 3.23.

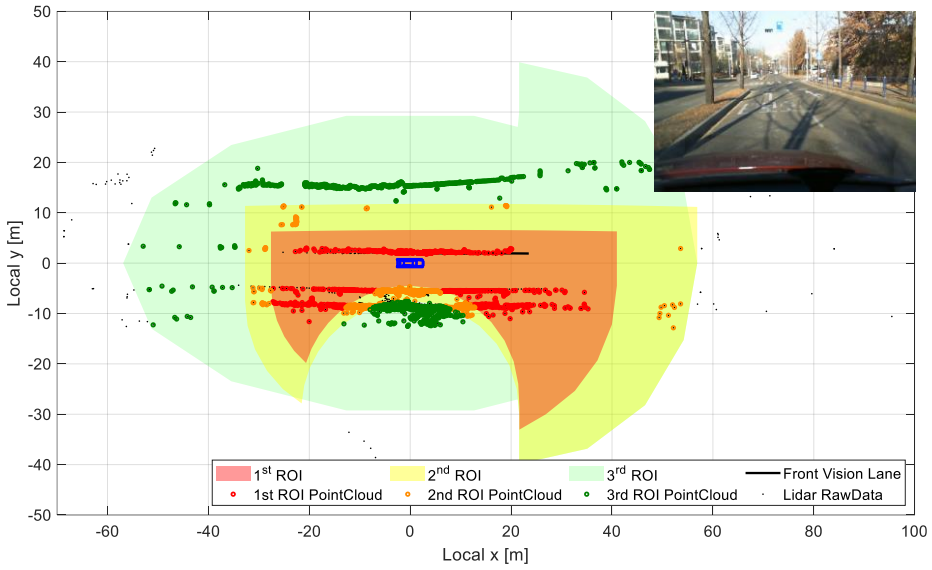


Figure 3.21. Point cloud categorization result (Normal driving).

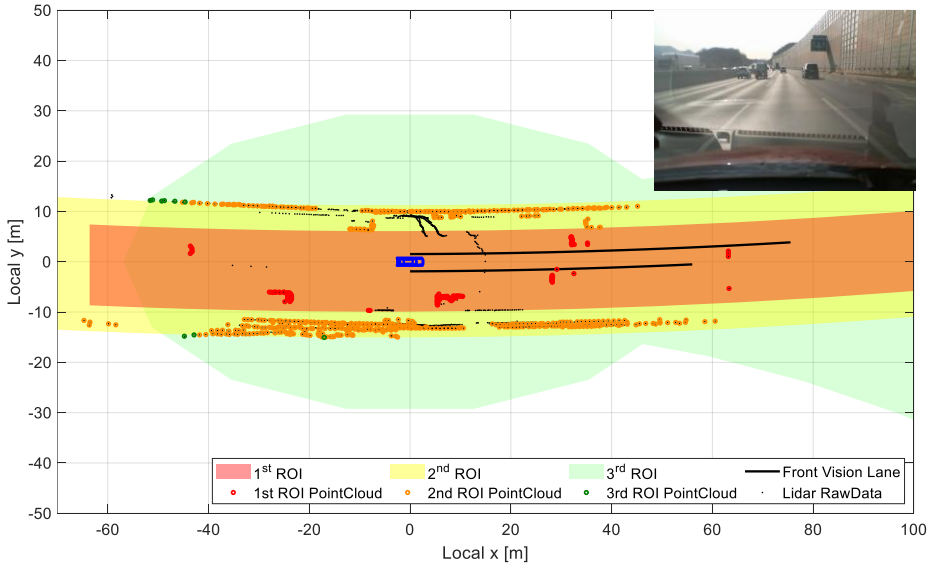


Figure 3.22. Point cloud categorization result (Lane change to the right).

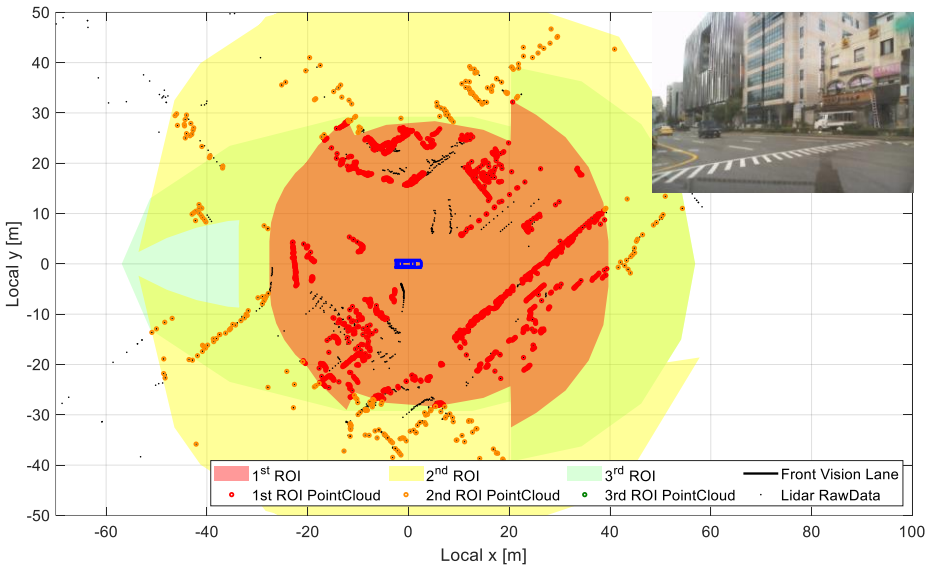


Figure 3.23. Point cloud categorization result (Intersection passing).

3.2.2. Separated Voxelization

Lidar's point cloud is very tightly spaced 3-D point cloud data, depending on the precision of the sensor. Since the raw data of the lidar point cloud, which includes sensor noise and surface, is very large, it should be downsized with minimal loss of environmental information to reduce computations. In general, a technique called voxelization, which defines the size of the smallest cell based on the Cartesian coordinate system, is applied. The voxelization consists of two processes:

1) Quantizing end-points of the beams: Considering the object point set O, the quantization of $X=(x,y,z)$, $X \in O$ is derived by following:

$$\breve{X}_j = \left\lfloor \frac{X_j}{ds_j} \right\rfloor \times ds_j \qquad (3.28)$$
$$where,\ \ j = \{x, y, z\}$$

where ds_j is the voxel size of j direction, here chosen as Table 9 and $\lfloor . \rfloor$ denotes the floor function. This process converts the original values in O to the quantized set $\breve{O}$.

2) Computing the occupancy cell values: The repeated elements in $\breve{O}$ denote points within the same cell. The occupancy value of a voxel is determined by counting the number of points in $\breve{O}$ that have the same value. The output of this task is a list of voxels with the occupancy values of $N\left(\breve{O} = U\right)$, $\forall \breve{X} \in U$, $\breve{X} = \left(\breve{x}, \breve{y}, \breve{z}\right)$, $\breve{X} \in U$ and $U = unique\left(\breve{O}\right)$.

The voxelization process, which performs the downsizing of the point cloud information, has a significant influence on the performance of the cognitive environment using lidar depending on the cell size setting. In general, most autonomous driving systems downsizing the same cell size for all point cloud information. As a result, the trade-off relationship between information loss and computational load occurs depending on the cell size. If the cell is too large, the loss of environmental information may occur, or if the cell is too small, the computational burden may be increased.

In this study, by applying a separate voxelization to each point cloud group

classified by ROI level through point cloud categorization, sufficient cognitive information to secure the driving safety of autonomous vehicles is secured. As defined above, the ROI level represents the surrounding area in order of importance according to the driving state of the subject vehicle. A high-priority 1st ROI is relatively significant in environment perception performance. Therefore, to minimize data loss to prevent collisions, the cell size is carefully set to use most raw data. Due to the relatively low importance in 2^{nd} and 3^{rd} order, the cell size was set to be large gradually, even if data loss is taken at some cost, to reduce the computational volume while maintaining the proper level of cognitive performance to increase efficiency.

When voxelization is applied, down-sizing is generally performed in units of cube cells. However, in this study, the axial direction was different. The density of point cloud varies according to the size of the voxel in each direction, which causes the error of object recognition performance. From the driving control point of view, the recognition error caused by the quantizing in the x-direction can be coped with by setting a safety margin for the x-direction voxel size. The quantizing error in the y-direction can cause inaccurate problems in estimating and predicting the width or position of the object, such as a vehicle, which has a significant impact on the planning of the behavior of the surrounding objects in terms of motion planning. Thus, to minimize this problem, a relatively dense voxel size setting than the x-direction is required. To prevent the loss of environment information, various voxel sizes are applied to find the appropriate size through simulations. As a result, the longitudinal voxel size is confirmed to maintain the perception performance when setting up to twice as much as

that of the lateral direction. At the 1st level, 5 centimeters is determined through experiments as a standard to obtain the downsizing effect while maintaining the ability to detect even pedestrians or small objects. At the 2nd level, it is necessary to recognize an object, such as a bike or motorcycle, with the potential to enter the small and 1st ROI area than moving faster than someone vehicles, taking into account the size of these objects was estimating the y-direction quantizing value to 20 centimeters. The 3rd level is less likely to affect driving immediately than the other areas. However, it is an area for continuous monitoring of objects such as vehicles with significant volume and dynamic capability, thus setting a reference value for the short length of the vehicle.

In this way, the y-direction voxel size was determined for each ROI level, and the x-direction voxel .size was defined accordingly. Besides, Figure 3.24 shows an example of the result of voxelization the point cloud classified according to ROI by level.

Table 9. 2-D cell size by adaptive ROI for separated voxelization.

Level of Adaptive ROI	Cell Size [m]	
	x	y
1st Level	0.10	0.05
2nd Level	0.40	0.20
3rd Level	0.80	0.40

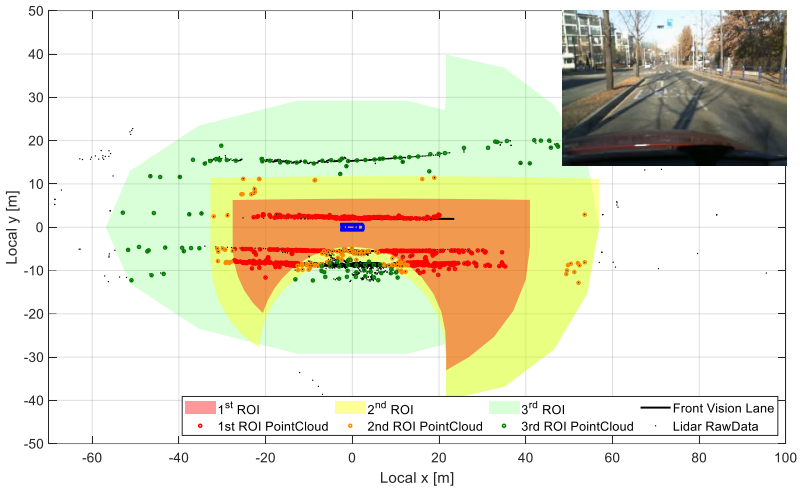


Figure 3.24. Separated voxelization result with left lane information.

3.2.3. Separated Clustering

Since the point cloud data represents the precise surface on which the laser beam is reflected, the boundary is accurately detected. However, in order to estimate the state variables for each object, the process of clustering the point cloud data must be preceded.

In the clustering process, graph-based clustering is conducted by applying a Euclidean Minimum Spanning Tree (EMST) technique. EMST is a spanning tree of a set of n points in a metric space, where the length of an edge is the Euclidean distance between a pair of points in the given point clouds. In this study, edge weights are calculated according to the definition of Euclidean distance, and breakpoints are determined. Then, a random sample consensus (RANSAC) technique is applied to distinguish the outlier between the edge weights. At the moment of detecting outlier, we determine to pass to the next

cluster is spanning tree concept. The derived cluster signal from the above procedure is used as a representative point of a detected object.

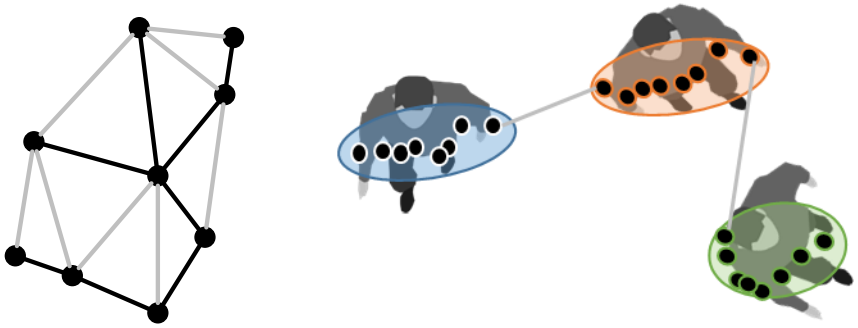


Figure 3.25. Scheme of EMST based clustering.

The clustering is typically performed on a point-to-point basis. Most environmental sensors, including lidar, acquire environmental information by transmitting and receiving signals radially. Because the vertical and horizontal angular resolution of lidar is fixed, the surface of an object that is close to the sensor, regardless of the size of the object, has a high density of point cloud distribution. On the contrary, the distance between adjacent beams increases as the distance from the sensor increases, resulting in a relatively low density of point cloud distribution.

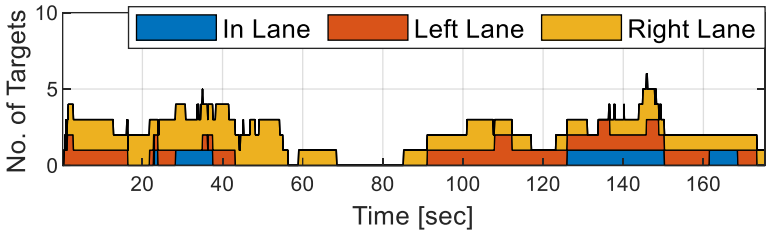
In Euclidean clustering, the performance of clustering is greatly affected by the distance threshold parameter, which is used to classify clusters, and the minimum point cloud number threshold, which is the size standard of lower limit point groups. The lower the distance threshold for precise clustering, the smaller the cluster are separated, resulting in a decrease in the state estimation

performance of the object because the same object is grouped into several parts according to the sensing geometry. The higher the distance threshold, the less likely it is to divide and cluster the same object, but there is a weakness that is difficult to distinguish when two or more objects are adjacent. Besides, the lower the minimum point cloud number threshold, the better the noise of the sensor is reflected in the clustering results, the higher the likelihood that the 로 cluster will fail to determine a signal with a low viscosity density as it is far from the sensor and object. In general, clustering with uniform parameters is applied to the entire point cloud, so clustering results are limited in properly reflecting the characteristics of the environmental signals obtained above. Therefore, to minimize the possibility of false recognition and maximize cognitive performance, a clustering strategy that reflects the geometric characteristics of the beam is required. Additionally, the voxel size of the voxelization process performed in the previous step also affects the point cloud density distribution and should be considered in the clustering process.

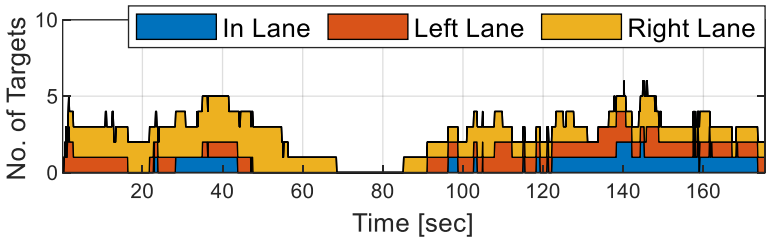
So far, we have defined the adaptive ROI based on the vehicle's driving status information, classified the point cloud data accordingly, and individually downsizing according to the area characteristics and importance. As an extension of this approach, clustering processing is performed by ROI to improve clustering performance in this subsection. The minimum point cloud number threshold is inversely proportional to the distance between the lidar sensor and the object to achieve proper clustering according to the density of the point cloud. Thus, the threshold setting was improved by reflecting the geometrical characteristics of each region.

The proposed algorithm verified through the data-based simulation. Since it is difficult to compare all cluster results distributed in all directions, clustering performance is compared for objects that must be considered in terms of driving safety. We compared the cognitive results of objects in in-lane, left-lane, and right-lane under normal driving conditions. The results of each clustering application are applied to the multiple object tracking algorithm described in Section 4.3 to obtain the results of the recognition of the surrounding objects. Conventional uniform clustering and adaptive ROI-based separated clustering are applied to the same driving data.

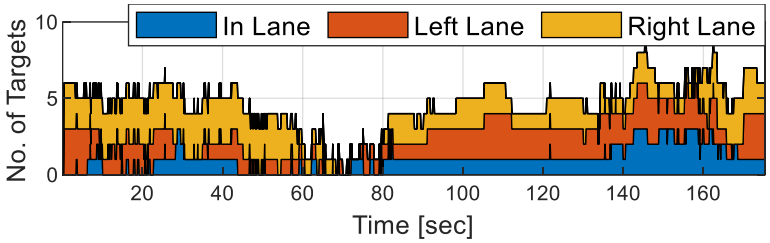
Figure 3.26 shows the number of valid targets for in-lanes and side lanes. There are a large number of recognized valid objects when the proposed separated clustering is applied throughout the simulation. It affects the prediction result based on the estimation of the state information of the object that can affect the driving of the vehicle, which has a positive effect on the motion planning performance to prevent collisions and the rapid response performance in case of accidents. Figure 3.27 compares the longitudinal distance to the most crucial front in-lane target in terms of control to avoid collisions. The conventional methods result in loss of in-lane target due to undetectable cluster loss in long-terminal resistance of approximately 75 m, but the proposed method can be found to be perceived as being up to approximately 120 meters.



(a) SOM based processing (max longitudinal range: 60m)



(b) SOM based processing (max longitudinal range: 120m)



(c) Adaptive ROI based processing

Figure 3.26. Comparison of the number of tracking targets.

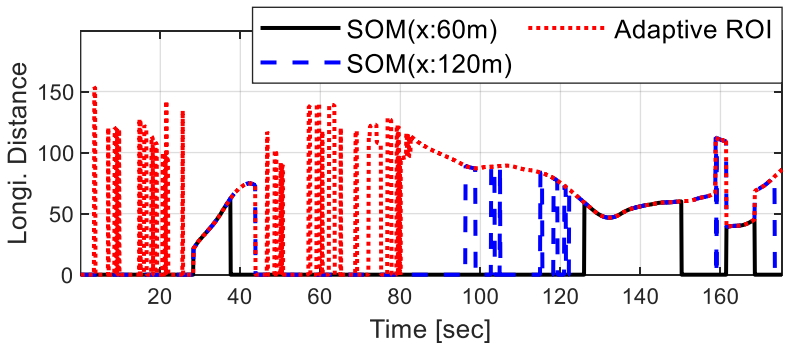


Figure 3.27. Tracking result of in-lane target by each perception algorithm.

Chapter 4 Environment Perception Algorithm for Automated Driving

Automated driving requires a quick and accurate ability of the environment perception. The environment perception algorithm computes environmental information for safe driving by processing signals obtained through various sensors mounted outward to the vehicle. It is essential to identify the characteristics of the target sensors and target systems and to reflect them so that no distortion or omission of information occurs. There can be two main types of environment information of self-driving cars: road facility information such as lanes and curbs, and objects necessary to prevent collisions in driving environments such as vehicles and pedestrians.

This chapter analyzes the characteristics of external environment sensors that are widely used in self-driving systems, and describes the environmental perception algorithms needed for the motion planning and control process by processing signals. All sensors and devices have latency in the process of sensing and transmitting information. These time delays occur in a series of processes related to information transmission, such as sensing, transmitting, and receiving, and the causes are complex. The time delay creates more distortion of the environmental information as the vehicle's behavior increases. Lidar sensor is widely applied to the automated driving system because it can precisely and accurately recognize the surrounding environment information compared to other sensors, but when information distortion occurs, it can cause

a decrease in perception ability. Therefore, in this study, we develop and verify the method to prevent information distortion of lidar information by analyzing and estimating the time delay characteristics of the target autonomous driving system and the mounted lidar sensor. Besides, an algorithm was proposed and verified to recognize the information on the road structures needed for the motion planning of the driving. Also, a method has been developed to estimate the status of dynamic objects that are essential to prevent collisions. In order to improve the limitation of model-based tracking (MBT), which determines and tracks the average model of a vehicle, the geometric model-free tracking (GMFT) is developed that can achieve tracking performance independent of the size or shape of an object by matching the point cloud information of object cluster. In particular, GMFT is applied to separated data processing based on adaptive ROI in Section 3.2 to recognize surrounding objects.

4.1. Time Delay Compensation of Environment Sensor

Nowadays, various kinds of automated driving systems (ADS) have been suggested and progressed in order to improve driver's safety. However, perceiving environment precisely is still challenging problem. The ADS have increasing demand for several sensor systems, which are not only complementary but also redundant. Much research has therefore been focused on high-level sensor fusion which requires reliable ability of each environmental sensor. Measurement of object with sensor such as laser scanner, radar, vision camera and so on, in driving area is crucial issue of ADS researches.

The time-delay is the major cause of debased performance and instability of ADS system. The network and sensor introduces delays in addition to process and transmission delay that are prevalent in most digital systems [Halevi,'88]. The signal handling should be designed to compensate for these delays. The laser scanners are widely used in field of engineering, such as autonomous vehicle driving, because of their advantages of non-contact measurement and high precision in over a large working range. This is the reason why various laser scanners are installed in most of automated driving vehicles [De Cecco,'06]. Thus, the laser scanner is focused on this research.

Besides, the system parameter uncertainties, it has been recognized that the time delay is also often the main cause of instability and poor performance of

systems [Malek-Zavarei,'87]. The filter design problems for uncertain time-delay systems have been studied [Pila,'99, Wang,'02, Wang,'04, Sahebsara,'07]. Up to now, in most relevant literature, the time delay of sensor is mostly assumed to be deterministic [Kaempchen,'03]. To work with sensor delay systems, normally, stochastic delay or uncertainty information are transformed to a stochastic parameter of the system.

Consequently, it is necessary to develop an algorithm which can minimize distance error due to time delay of measured object in driving area. To accomplish this task, a time delay of the laser scanner is analyzed and a delay compensation algorithm has been developed. The proposed algorithm was devised based on the ideas of forward estimation of movement state in posterior parietal cortex [Mulliken,'08]. The widely used transformation can be used to the model with a maximum of the one sampling delay, while the new proposed representation in this seection. The proposed delay compensation algorithm was verified via automated driving system in the open loop simulation.

4.1.1. Algorithm Structure of Time Delay Estimation and Compensation

To improve driver's safety, more accurate environment perception is required. Due to processing latency of laser scanner, delay compensation is needed. Therefore, we developed the forward estimation of object algorithm as shown in Figure 4.1. The laser scanner data and ego vehicle's chassis signals are used to compensate the sensor latency. Consequently, the proposed algorithm can lessen the sensor latency and modified object signals can be obtained.

Figure 4.2 describes the proposed compensation algorithm scheme. The

coordinate transformation using vehicle state obtained from vehicle filter is applied in order to correct the delayed sensor data.

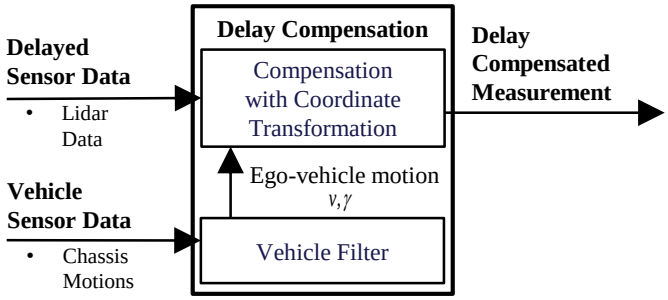


Figure 4.1. Block diagram of delay compensation process.

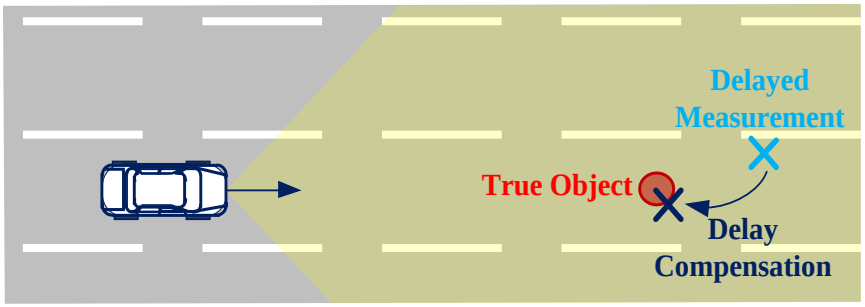


Figure 4.2. Scheme of delay compensation.

For the verification of the algorithm, vehicle tests in real road have been conducted. The proposed algorithm is implemented on test vehicle to confirm the process delay effect on automated driving.

4.1.2. Time Delay Compensation Algorithm

The vehicle's current dynamic states are estimated with the Kalman filter. Then, the forward estimation is applied to object by using coordinate

transformation. The states of subject vehicle obtained from vehicle filter can be used in forward estimation.

4.1.2.1. *Vehicle Filter*

The Kalman filter is used to estimate present vehicle states like as longitudinal velocity, yaw rate, longitudinal acceleration and yaw acceleration from the vehicle sensor signals such as steering angle, yaw rate, longitudinal velocity and longitudinal acceleration under the assumption of the Gaussian white noise. As previously stated, the state vector x is defined as following in order to represent the driver's intention and the vehicle's planar behavior:

$$x = \begin{bmatrix} v & \gamma & a & \dot{\gamma} \end{bmatrix}^T \tag{4.1}$$

where v is the longitudinal velocity, γ is the yaw rate, a is the longitudinal acceleration and $\dot{\gamma}$ is the yaw acceleration. The measurement vector is defined as following to reflect the available sensor information.

$$z = \begin{bmatrix} v & \gamma & a & \delta_f \end{bmatrix}^T \tag{4.2}$$

where δ_f is the front wheel steering angle. Assumption of the time derivatives of the longitudinal acceleration and the yaw acceleration can be considered as process noise, the process model and measurement model are given by following form:

$$x[k+1] = F[k] \cdot x[k] + w[k] \tag{4.3}$$

$$z[k] = Hx[k] + v[k] \tag{4.4}$$

where

$$F = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & h_{42} & 0 & h_{44} \end{bmatrix}$$

$$h_{42} = \frac{2l_f{}^2 C_f + 2l_r{}^2 C_r}{2l_f C_f v} \qquad h_{44} = \frac{I_z}{2l_f C_f}$$

where Δt is the sampling time which taken as 0.1 second in this study, I_z is the yaw moment of inertia, C_f and C_r are the front and rear wheel cornering stiffness, respectively l_f and l_r are the distances from vehicle's center of gravity to front and rear axles. Two elements in 4th row of measurement matrix are determined from the bicycle model which is most well-known lateral vehicle dynamics model [Rajamani,'11].

The process noise is assumed to be a white noise with associated covariance matrix, W. The measurement noise is also assumed to be a white noise with associated covariance, V. Note that measurement model, H, is time varying because there exist longitudinal velocity in the element of the matrix. Therefore, it should be re-calculated at each time step. With above process and measurement model, vehicle states are recursively estimated by using the Kalman filter which is a sequence of time and measurement update steps as following specific equations [Simon,'06]:

Time update

$$\bar{x}[k] = F \cdot \hat{x}[k-1] \tag{4.5}$$

$$M[k] = F \cdot P[k-1] \cdot F^T + W \tag{4.6}$$

Measurement Update

$$\hat{x}[k] = \bar{x}[k] + K[k] \cdot \left(z[k] - H[k]\bar{x}[k] \right) \tag{4.7}$$

$$K[k] = M[k]H[k]^T \cdot \left(H[k]M[k]H[k]^T + V \right)^{-1} \tag{4.8}$$

$$P[k|k] = \left(I - K[k]H[k] \right) \cdot M[k] \tag{4.9}$$

4.1.2.2. *Forward Estimation*

In this study, forward estimation algorithm is proposed in order to correct object information as described in Figure 4.3.

Under existence of process delay, the host vehicle moves for time period of sensor delay. Consequently, estimation of the previous host vehicle position that is the exact time of sensing is possible to diminish inaccuracy.

The coordinate systems of host vehicle and object and the relations of them are defined as described in Figure 4.4. The current host vehicle frame is determined as $\{1\}$ and past host vehicle frame is defined as $\{0\}$. Also, the frame of true object which measured in driving area was decided as $\{2\}$ and the frame of delayed measurement is determined as $\{3\}$. The relation T_{10} indicates coordinate transformation of host vehicle during delayed period and

T_{32} presents frame transformation between true and delayed measurement. In this study, we assume that T_{32} is nearly same as T_{10} because the delay effect results from vehicle movement for the delay period.

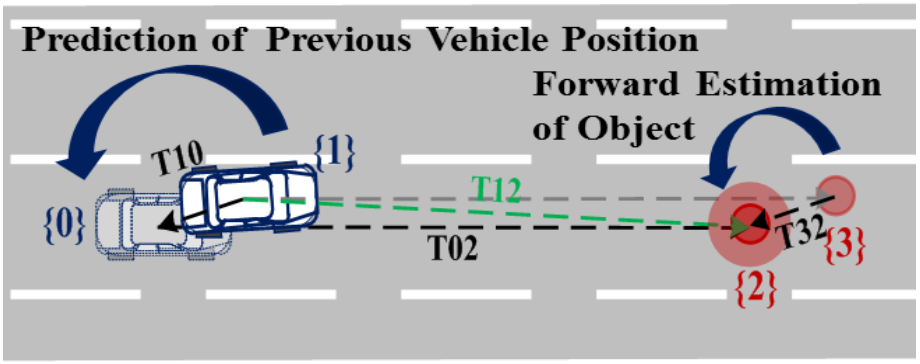


Figure 4.3. Schematic description of forward estimation.

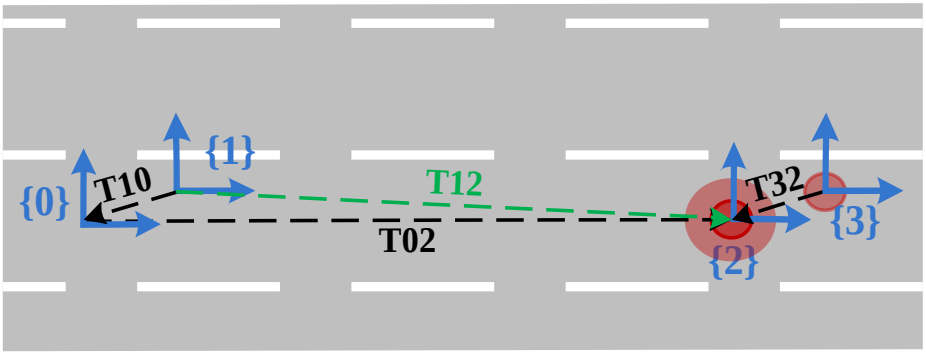


Figure 4.4. Definition of coordinate systems and their relations.

The measurement vector is defined as following.

$$x = \begin{bmatrix} p_{x,target} & p_{y,target} & 1 \end{bmatrix}^T \tag{4.10}$$

where $p_{x,target}$, $p_{y,target}$ are the object position relative to ego-vehicle.

The compensated delay signal $\hat{x}_{compensated}$ is obtained multiplication of transition matrix T_{32} and delayed measurement $x_{delayed}$.

$$\hat{x}_{compensated} = T_{32} \cdot x_{delayed} \tag{4.11}$$

where,

$$T_{32} = \begin{bmatrix} \cos(\frac{\gamma \cdot \Delta t}{N_{DR}}) & -\sin(\frac{\gamma \cdot \Delta t}{N_{DR}}) & v \cdot \frac{\gamma \cdot \Delta t}{N_{DR}} \\ \sin(\frac{\gamma \cdot \Delta t}{N_{DR}}) & \cos(\frac{\gamma \cdot \Delta t}{N_{DR}}) & 0 \\ 0 & 0 & 1 \end{bmatrix}^{N_{DR}}$$

where Δt is process delay assumed, γ is yaw rate and v is longitudinal velocity of host vehicle, and N_{DR} is the split number of dead reckoning process which is taken as 10000 in this study.

4.1.3. Analysis of Processing Delay

First of all, the laser scanner's processing latency has to be analyzed. Specification sheet of sensor provide us only sampling frequency, therefore we attempt to get its processing time-delay by conducting actual experiment.

4.1.3.1. *Configuration of the Test Vehicle*

The test vehicle as shown in Figure 4.5 is used for delay analysis and

verification of proposed algorithm. Laser scanner, radars, camera and vision sensor are installed on test vehicle to perceive driving environment. Also, auto box, ECU and PC are mounted in order to control ADS. Additionally, DGPS is also used to this test to determine reference object position. In this research, the 4-layer laser scanner located in front of test vehicle is concerned. The proposed algorithm of this research is applied and tested to this vehicle in order to verify the process and transmit delay effect on automated driving situation.

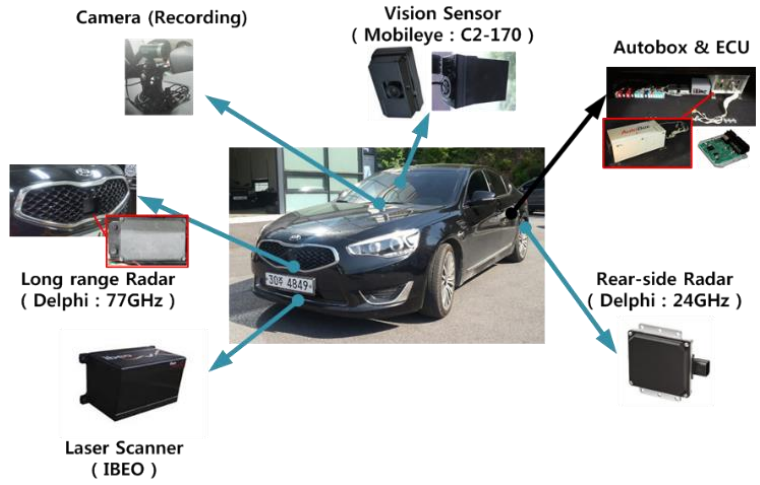


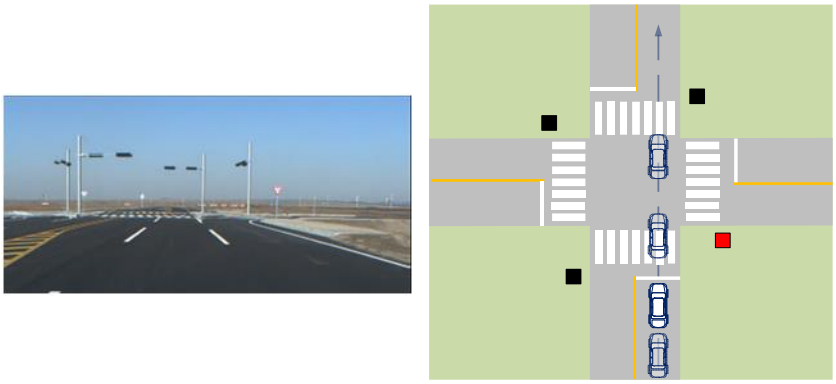
Figure 4.5. Sensor configuration of test vehicle.

4.1.3.2. *Test Scenario*

The vehicle tests have been conducted at the intersection of Intelligent Transportation System (ITS) test road in Korea Transportation Safety Authority. Descriptions of this test situation are given in Figure 4.6.

The place of experiment is depicted in Figure 4.6-(a) and the specific test

scenario is described in Figure 4.6-(b). The test site is an intersection which has four traffic lights in each corner. In order to obtain the ground truth of given obstacles, the test vehicle measured the positions of traffic light poles while standing still at the stop line for 1 minute. The accumulated signal data at this process is assumed the actual true position of object. Secondly, the same poles under driving state are measured. By comparing these data obtained from the experiment, time delay of laser scanner can be investigated.



(a) Test road condition (b) Test scenario

Figure 4.6. Test Situation Description of Intersection with traffic lights.

4.1.3.3. *Test Result*

From the test, time delay of sensor can be analyzed by comparing the position of four traffic poles on the intersection. The test result is shown in Figure 4.7. In Figure 4.7, the black circles which are obtained with halted vehicle located in intersection indicate reference position of traffic light poles. And the blue triangles represent the same pole signal on driving pass through the intersection

with 15kph vehicle speed. The measurement while driving was perceived as far object that are compared to true position. The difference between true object and delayed measurement is approximately 0.4 meters as shown in Figure 4.7. Due to 15kph vehicle velocity, we can roughly expect that around 100ms latency exists.

From the result, when the vehicle is moving, the system measures the delayed objects which are far away from the true position. It means that distance error occurs due to time delay. Also, we can expect that the distance error of each object is proportional to relative motion between host vehicle and object.

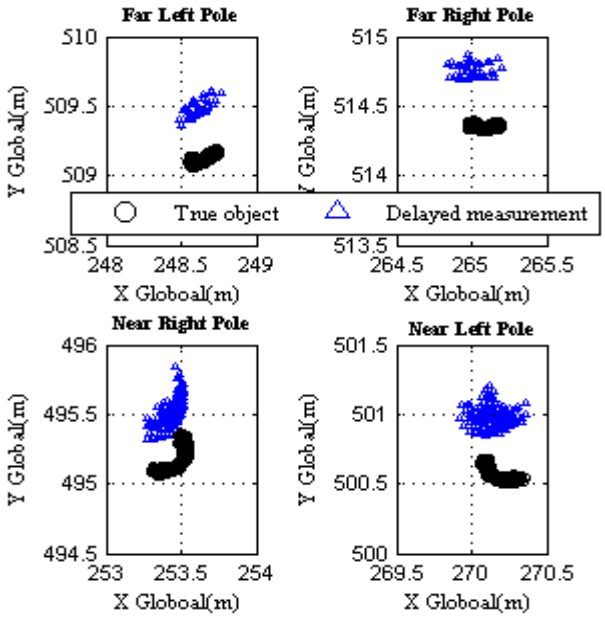


Figure 4.7. Comparison of global position of poles.

Besides, experiments are conducted with different host vehicle speed in

0~30kph range as shown in Figure 4.8. The four cases of driving data are used for analysis. In Fig.8, the red diamond and magenta x marks indicate the distance error when test vehicle passes through the intersection with waiting in front of stop line. On the contrary, blue square marks and green triangle marks describe the distance error while passing through intersection with no stops under 15kph and 26kph. In Figure 4.8, we can find out that tendency that the distance error increased in proportion to relative velocity between host vehicle and obstacle. Also, the distribution of y-axis is assumed as the effect of sensor noise and obstacle width.

If delayed measurement is used as an object information, the safety clearance becomes unreliable due to distance error. To secure driver's safety, sensor's process delay has to be estimated and compensated.

The estimations are attempted under various process latency assumptions. The black circles which indicate true position of object and the blue triangles which represent delayed measurement under driving condition are marked in Fig.9. What one has to look at is the red square marks which represents the result of forward estimation. We applied 0.1s delay assumption in Figure 4.9-(a) and also 0.15s in Figure 4.9-(b). Accordingly, it is easily noticed that forward estimation improves perception accuracy.

The error between true and compensated object was defined as follows.

$$\min_{\triangle t}\left[J = \frac{1}{N}\sum\sqrt{\left(\hat{x}_{compensated} - x_{true}\right)^2 + \left(\hat{y}_{compensated} - y_{true}\right)^2} \right] \qquad (4.12)$$

where x_{true}, y_{true} are the position of true object, $\hat{x}_{compensated}$, $\hat{y}_{compensated}$ are

the position of compensated object and N is the number of object measurement.

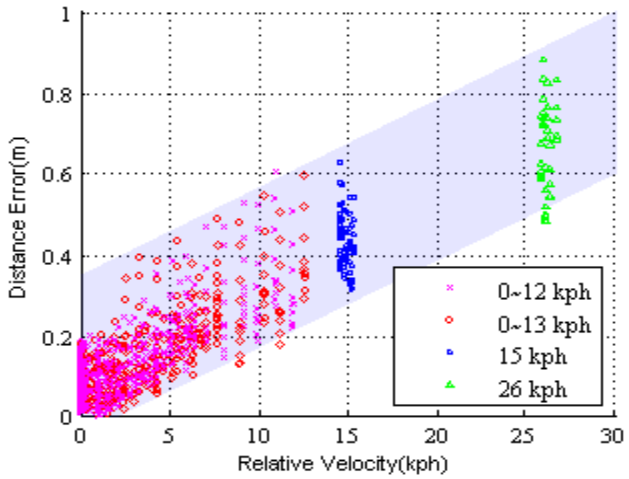
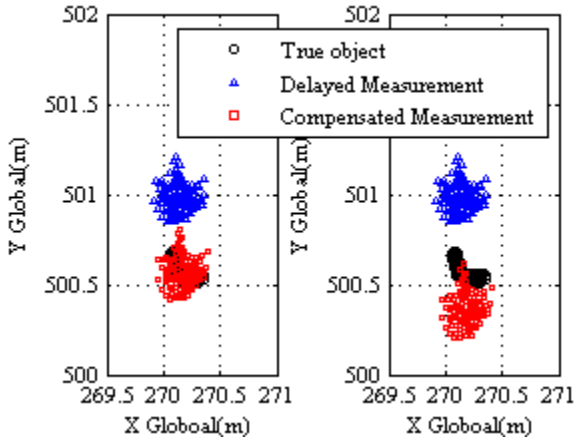


Figure 4.8. Measured global object position with 0~30kph range.



(a) Assumed 0.1s delay (b) Assumed 0.15s delay

Figure 4.9. Compensated signal of forward estimation with $\triangle t$ =0.1s, 0.15s.

The assumed delay Δt, which minimize the error J, is the actual process and transmission delay of laser scanner. The error, J, is calculated by 0.001sec time interval as shown in Figure 4.9. Therefore, the minimum error occurs at 0.112s.

From this result, the time delay of laser scanner was estimated as 0.112s and then, Δt is determined as 0.112s to compensate. The error probability is calculated in Figure 4.10. The distance error is highly decreased by applying compensation under delay 0.112s. This is the reason why assumption of 0.112s time delay is reasonable.

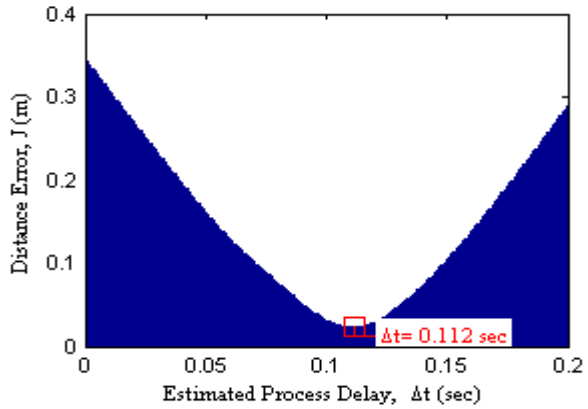


Figure 4.10. Calculated error J by Δt.

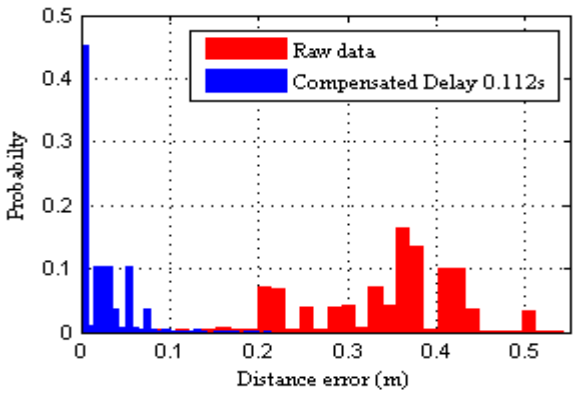


Figure 4.11. Comparison of distance error probability.

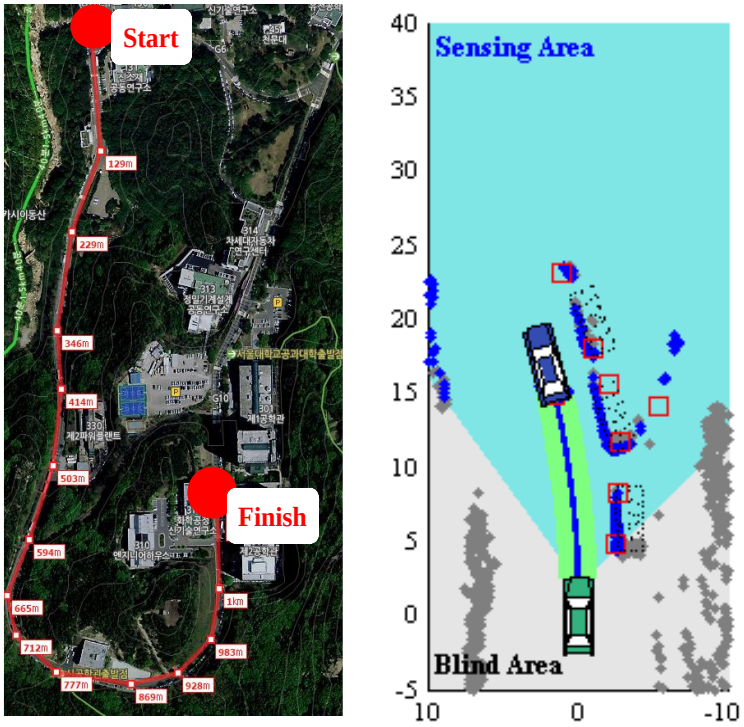
4.1.4. Test Data based Open-loop Simulation

In previous section, the processing delay of our laser scanner which minimizes the distance error is derived as the value of 0.112s. The effect of presence of process delay compensation in vehicle control should be investigated. Therefore, test data based open-loop simulation has been conducted to validate the automated driving performance enhancement due to the proposed time delay compensation algorithm.

The test data have been acquired for several automated driving tests at the internal road of Seoul National University as depicted in Figure 4.12 (a). The scheme of automated driving strategy is shown in Figure 4.12 (b). The system detect the preceding vehicle and perform the following driving with ensuring the safety distance from the measured obstacles.

The given test roads have quite complicated environments to drive automatically. The lanes were hard to be distinguished because of faded paint. There are a lot of buses parked along the road as shown in Figure 4.13 (b). In Figure 4.13 (c), there exist non-vehicle obstacles such as pedestrian and guardrail to avoid. Also, we need to consider other traffic participants like oncoming and preceding vehicles as described in Figure 4.13 (d).

The open loop simulation was conducted via logged data of our test vehicle by using MATLAB. In this simulation under open loop, our system calculated desired local path and steering angle to drive automatically. By comparing results which the proposed algorithm is applied and not applied, the effect of delay compensation will possibly confirmed.



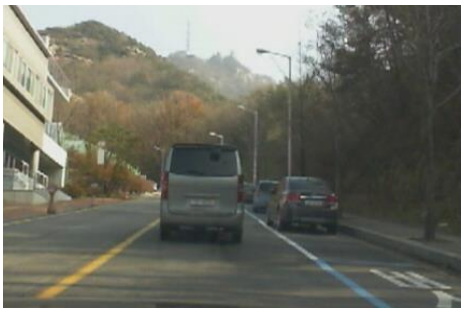
(a) Test road in SNU (b) ADS configuration

Figure 4.12. Description of test scenario.

(a) Inside of test vehicle

(b) Highly extreme condition

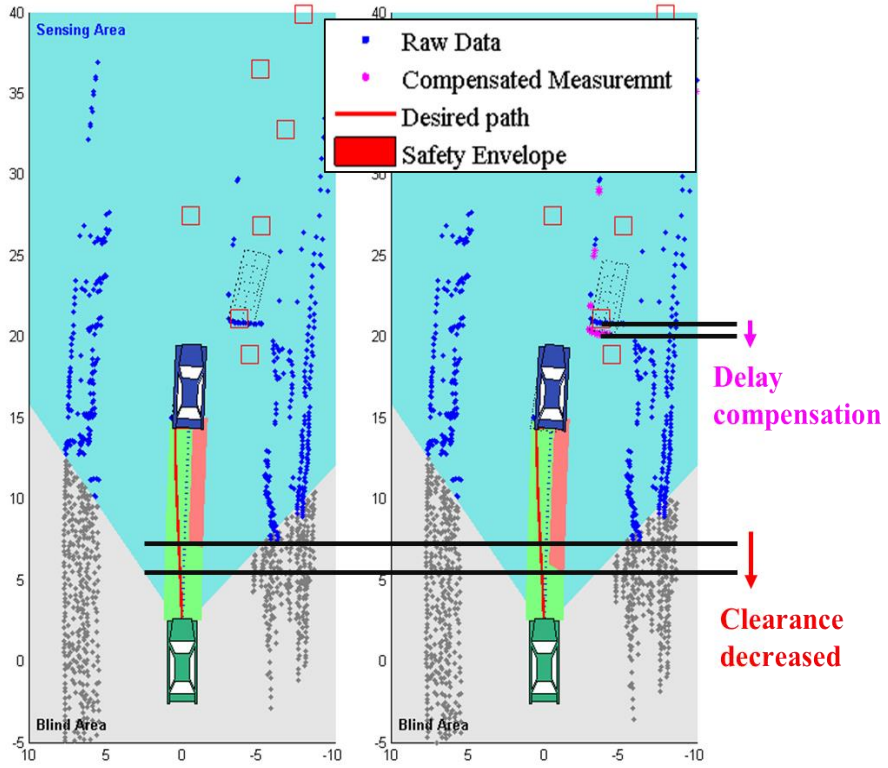
(c) Non-vehicle obstacles



(d) Other traffic participants

Figure 4.13. Major scenes of test environment.

From this simulation, desired steering angle is obtained while the proposed algorithm is applied to the automated driving system. The comparison plot of desired steering angle is depicted in Figure 4.15. In Figure 4.15, around 100 and 108 seconds in horizontal time axis, it is confirmed that phase of desired steering commend has been moved forward due to the proposed compensation. When the forward estimation applied to the object signals, most of compensated signals are located closer to the host vehicle. Because of these compensation, safety clearance decreased and the system need earlier control to grantee the safety margin. Details at this critical situation are described in Figure 4.14 (b).



(a) No compensation (b) Compensated 0.112s delay

Figure 4.14. Scene comparison between non-compensation and applied compensation.

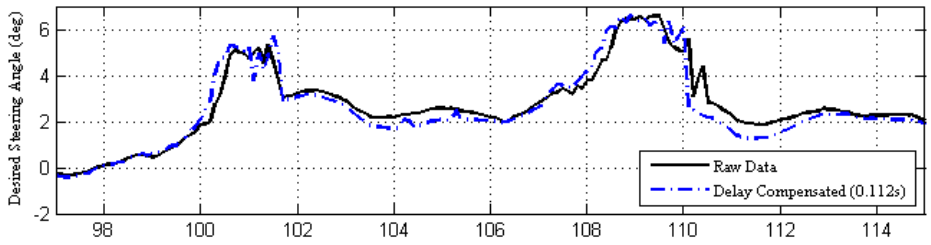


Figure 4.15. Comparison of desired steering wheel angle.

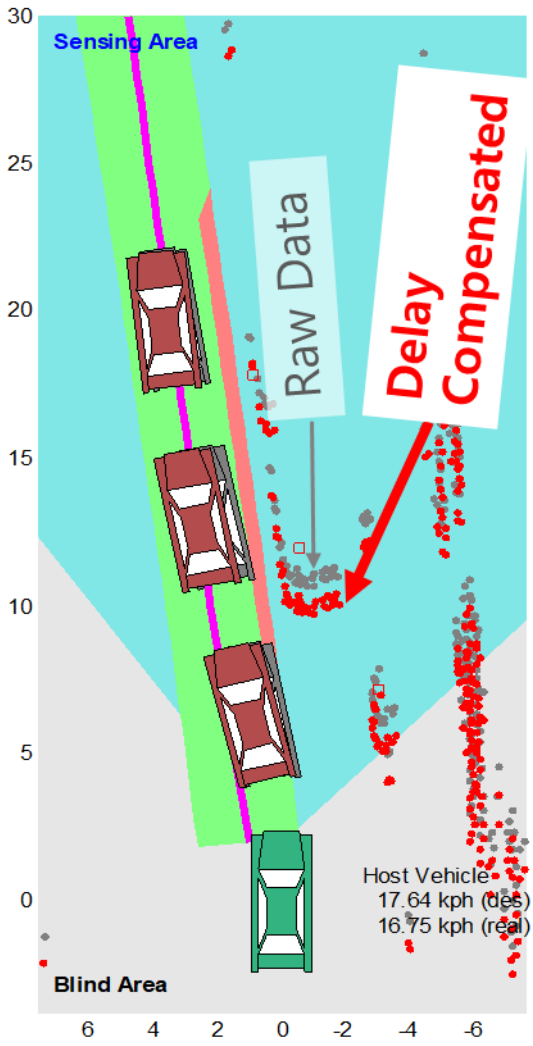


Figure 4.16. Change of predicted vehicle motion with delay compensation.

4.2. Environment Representation

Environment information that should be detected from an automated driving perspective consists largely of two types: road facility, such as lane, marker, road boundary, and surrounding objects information. Using lane and road boundary features, the pathways depending on the driving environment can be determined, and the route planning method is implemented to recognize the surrounding objects to prevent collision. This section proposes a detection algorithm for lane and road boundary information and static obstacles that should be preceded by motion planning for sensor-based automated driving.

In general, it is assumed that the vehicle runs on the road. For this reason, a person determines the desired driving path of a vehicle by recognizing the facilities indicating the lanes and road boundaries such as road surfaces, lanes, and markers through vision. The route information to be driven through painted lanes on the road can be found. In the case of motorway, a guardrail or a median strip can be a kinds of road boundary. In the case of an urban road, the curb that separates sidewalk and road can be regarded as a road boundary. Because lanes and road boundaries are largely regular and continuous on most roads, this information can be used to determine the desired path of vehicle. In addition, a static obstacle map is constructed to utilize information on the stationary objects that are not standardized road facilities but still in the path planning. In this section, we propose lidar based detection algorithms of lanes, road markers, and road curbs, and static obstacle map construction approach. The proposed

algorithm is verified through experiments on roads that represent complex urban environments.

Furthermore, the road information extracted in this section is a fixed facility that does not change unless artificial modifications are made, so it is included in the element of the high-definition map (HD map) currently constructed in the main section. Therefore, detected road facility information can be applied not only to route planning of sensor-based self-driving but also to Vehicle localization through map matching.

4.2.1. Static Obstacle Map Construction

In the boundary extraction process for static objects, we apply a grid representation technique to confirm the signals that become an actual obstacle. Grid representation technique means dense raw data transformed into a representative grid cell described in Figure 4.17. Since raw static signals are large as inefficient in view of the memory system, data downsizing through grid representation is required to improve computational efficiency.

A grid is fixed as a square, and representative grid points indicate that object boundaries can be expressed as configured grid size. Each grid of the map has the counter, which indicates how many points existed in that grid cell. If the counter value of the cell is higher than a certain designated threshold, the grid is considered occupied by the object. The set of occupied cells is considered a static obstacle map. To cope with the noisy measurement of lidar and guarantee minimum space in any case, a safety margin is determined by considering the sensor's systematic error, statistical error, and designated minimum clearance.

In the case of the candidates for moving objects, they are detected by comparing the current observation with estimated moving objects from MOT to be described in Section 4.3. If a point is entirely within the area of the estimated moving object, the point is considered as a part of the moving object.

An example result of static obstacle map construction is shown in Figure 4.18. The sensing area of lidar is depicted as a cyan-colored region. Black points are the current observation of lidar, green squares are estimated static objects, and blue circles are moving objects. Also, red filled squares are the occupancy grid of static obstacles, and blue filled squares are the occupancy grid of moving

objects. It can also be seen that the performance of moving object estimation is sufficient to construct static occupancy cells.

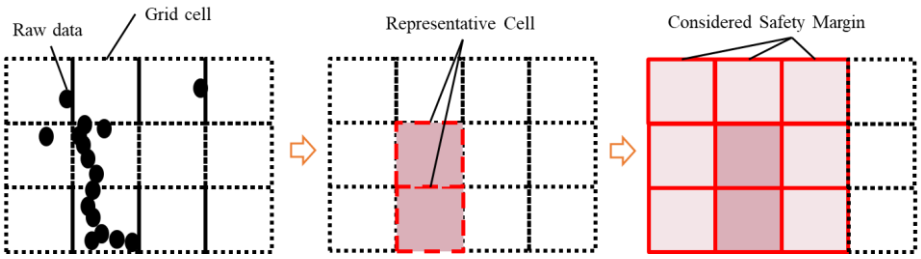


Figure 4.17. Transformation point cloud data into representative cell.

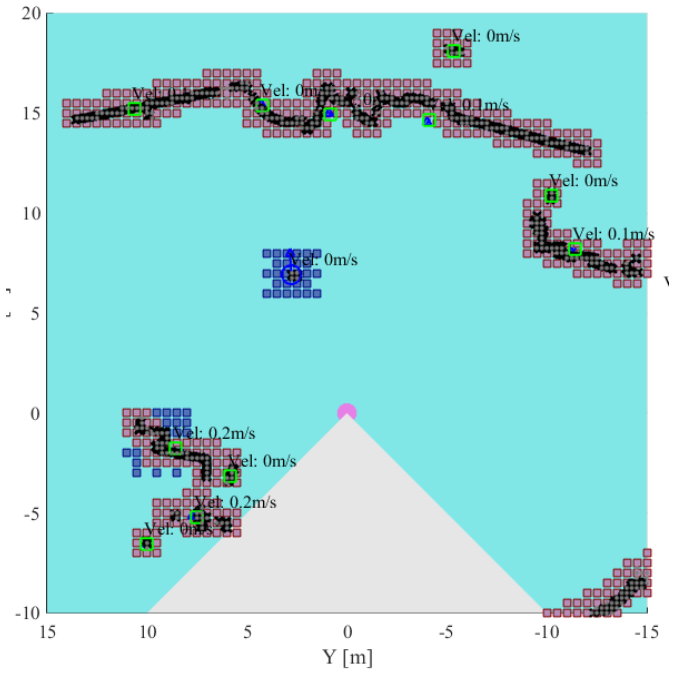


Figure 4.18. Example result of static obstacle map construction.

4.2.2. Lane and Road Boundary Detection

It is essential to recognize the road boundaries such as lanes and curbstones for collision-free driving in a narrow and complex urban environment. Road boundary information cannot only be used directly to establish a sensor-based desired driving path, but also to be used in the vehicle localization through map-matching. Many researches [Schreiber,'13, Tao,'13], use lane marking data to build their digital map. The second most used feature is a curb [Schreiber,'13, Hata,'14b]. Curbs usually appear at the borders between streets and sidewalks. Therefore, this section presents ways to extract lane and curb information using lidar.

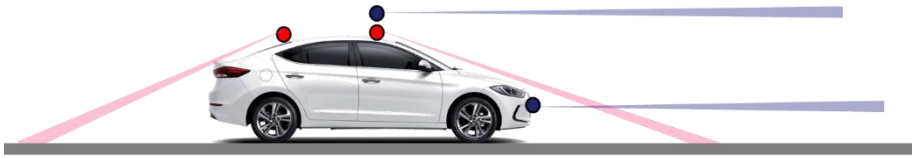
4.2.2.1. *Lidar based Curb and Lane Detection*

The lane marks close to the own vehicle can be reliably extracted by their corresponding intensity value of a lidar measurement [Isogai,'09]. The effect that dark road surface reflects significantly less laser light energy than the brighter lane marks was taken advantage of in that work to extract lane mark measurements by using an adaptive threshold on the intensity channel of the lidar sensor.

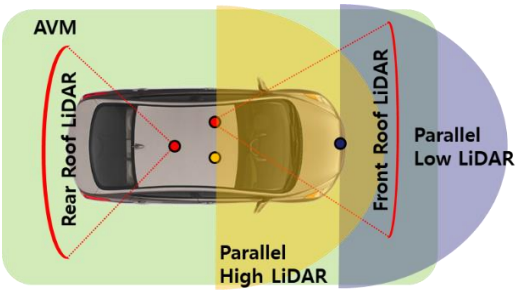
In this research, four SICK LMS511-10100 are installed on the test vehicle. To detect lane marks on road surfaces, two lidar are located on the roof of the vehicle as depicted in Figure 4.19.

The lidar layers are facing ground and are intersecting the road surface in distance of approximately 3 meters for the front bumper and 2 meters for the rear end of vehicle. Figure 4.19 shows two lidars we use for curb and lane

detection in red color circle and their view angle toward down. LMS511 provides 190 degree field of view range with 0.25 degree angular resolution and its scan rate is up to 25 Hz. Their sensing maximum range reaches 80 meters with a 50 mm error and they also can output reflectivity values which called Received Signal Strength Indicator (RSSI). RSSI is the measurement of power received by the lidar. This value is generated for every measurement and has an arbitrary unit with a logarithmic characteristic.



(a) Side view of four lidar sensors installation



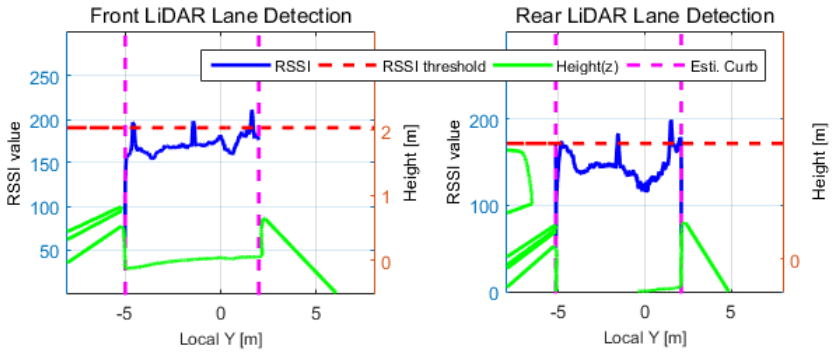
(b) Top view of four lidar and AVM

Figure 4.19. Sensor configuration of test vehicle to detect curb and lane.

First of all, each scan is processed individually. In Figure 4.20, where scan signal points are projected in the vehicle local coordinates in green color, it can be noticed that the center of the road is quite flat. Then, we tried to found a

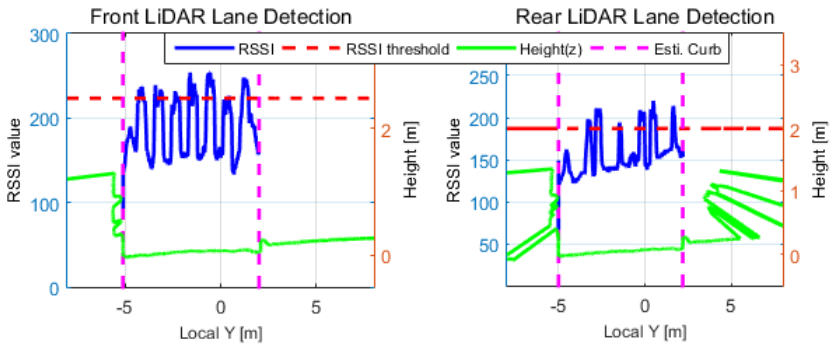
polyline expression that best approximates the road scan data. By using Random Sample Consensus (RANSAC) algorithm, only two thresholds are necessary; the first one indicates the expected outlier rate in the points set, which is directly related to the iterations number, and the second one specifies the distance above which a point is considered as an outlier. The result of road boundary estimation which can be called as curb stone is presented in Figure 4.20. Vertical magenta dashed line represent the extracted curb stone point using the RANSAC algorithm. It can be seen that the road is well extracted.

With a correctly extracted ground and using the reflectivity data, we can also extract lane marks through a simple thresholding. Asphalt presents a much lower reflectivity than road marks so that threshold determination is quite easy [Dietmayer,'05]. Nevertheless, this approach can in some cases be less robust than image processing methods, as it highly depends on marking reflectivity, which is faster deteriorated than white painting. In Figure 4.20 and Figure 4.21, parallel red dashed line indicates adaptive RSSI threshold and blue dots are RSSI values of the road. In these figures, blue points which are exceed the RSSI threshold represent the extracted points of lane marks. As shown in Figure 4.20, it can be easily found that three points exceeding RSSI threshold are left, center and right lane of two-lane road. Also, Figure 4.21 shows the pattern painted on the crosswalk can be extracted.



(a) Front lidar (b) Rear lidar

Figure 4.20. Lidar based lane detection result using RSSI (Normal Road).



(a) Front lidar (b) Rear lidar

Figure 4.21. Lidar based lane detection result using RSSI (Crosswalk).

4.2.2.2. *Test Results*

Vehicle tests were conducted at the driver's license test course located in Incheon as show in Fig.8. The length of the designated path is about 600 meters long. The vehicle passed 90 degree course, S-curve course, and three intersections. The width of normal road is generally 3.5 m, but averagely 3.0 m

in target route. This is the reason why high accurate level of detection ability is fundamentally required to avoid collision accident of automated driving in complex urban road.

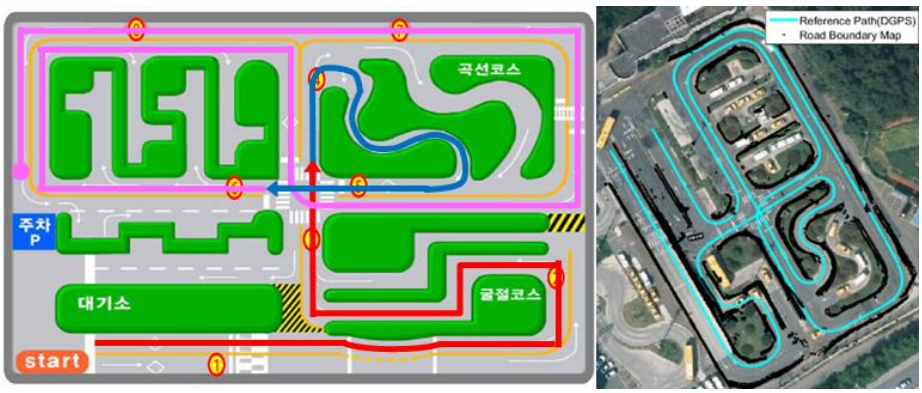


Figure 4.22. Test site description of lane and curb stone detection.

The road boundary and lane detection algorithm has been verified by real time automated vehicle on designated route. Figure 4.23 and Figure 4.24 show algorithm verification scenes during real-time vehicle test. In these figures, the blue and cyan dots express detected road boundaries with accumulation, and magenta and red dots indicate the extracted road lane with accumulation.

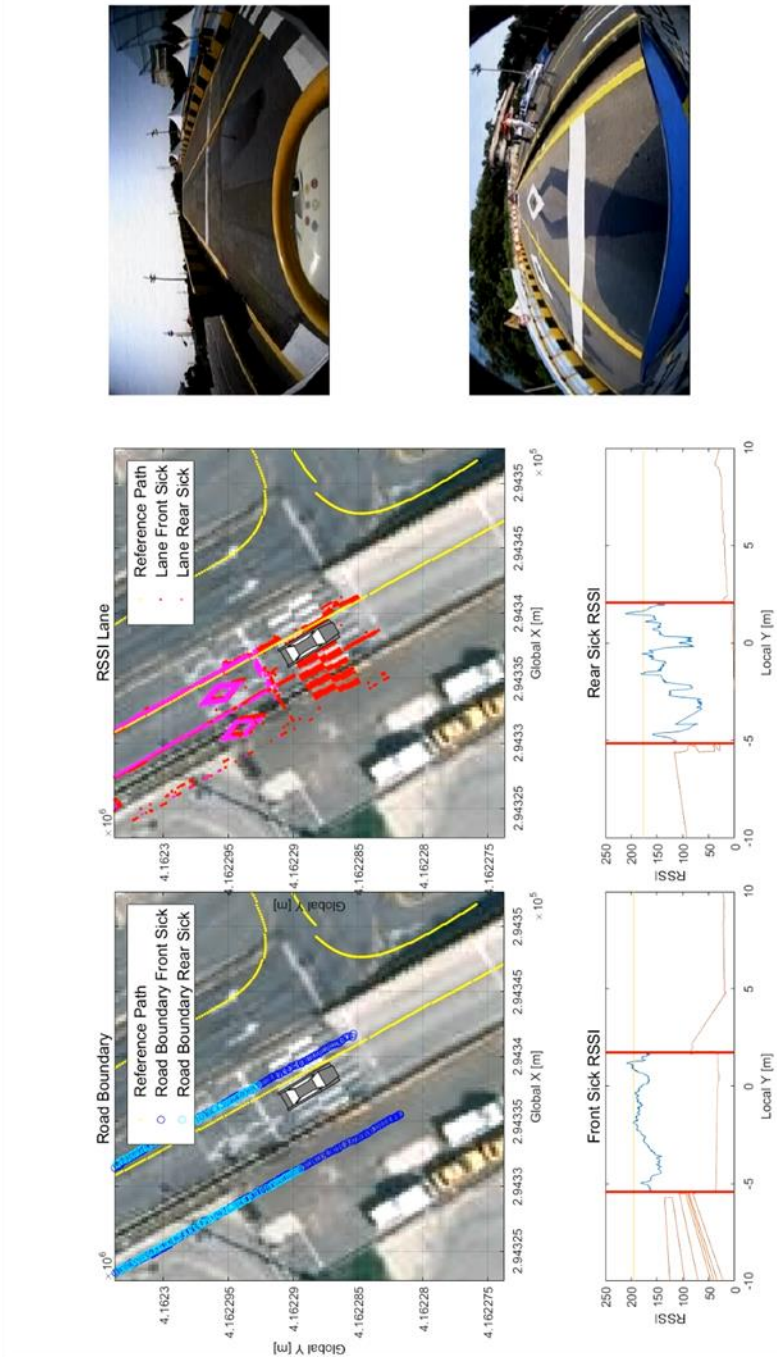


Figure 4.23. Vehicle test result of lane and curb stone detection (Crosswalk).

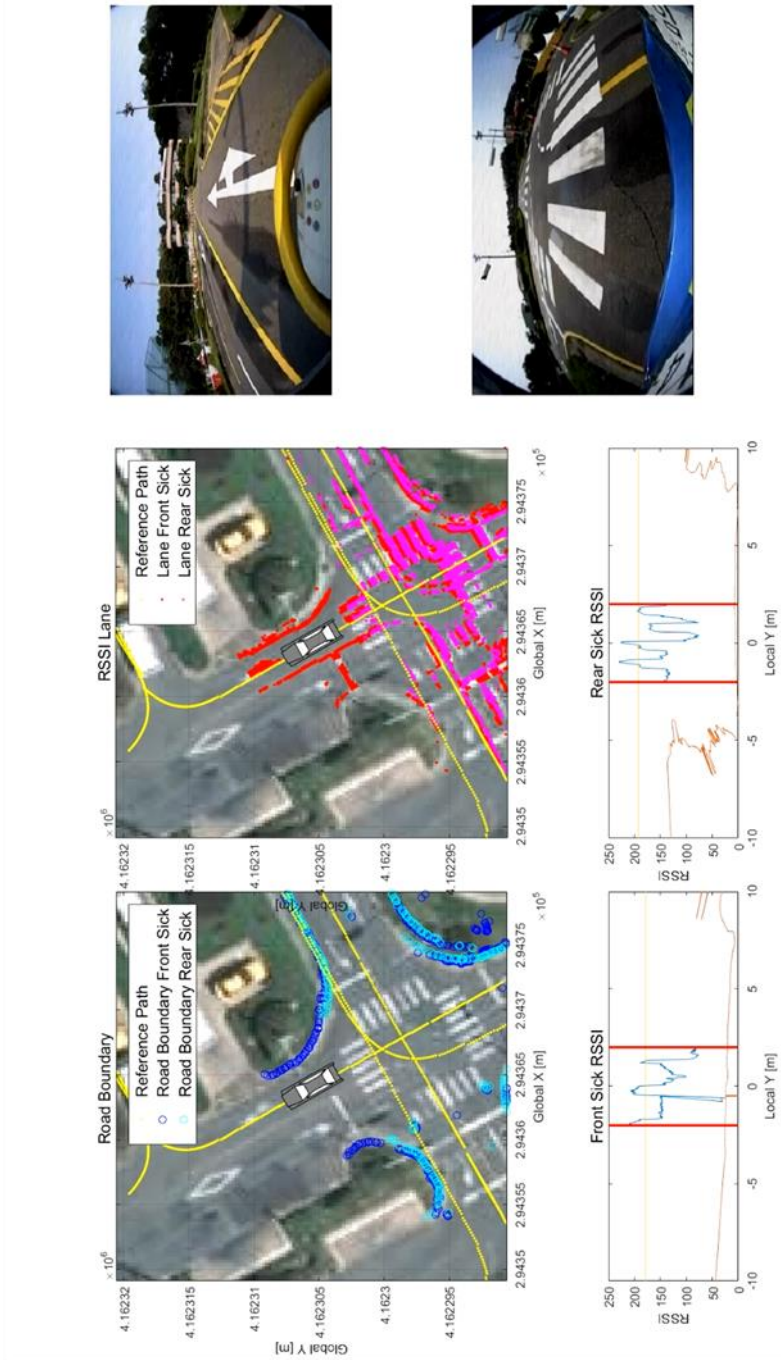


Figure 4.24. Vehicle test result of lane and curb stone detection (Intersection).

4.3. Multiple Object State Estimation and Tracking based on Geometric Model-Free Approach

In this study, a novel approach is applied to enhance the performance of detection and tracking of moving objects (DATMO) by using a Geometric Model-Free Approach (GMFA) on our ADS system for real-time application. The proposed algorithm detects objects and estimates their states irrespective of feature shapes. The following are the major contributions of this study.

▶ the proposed approach is robust against sparse point clouds, object shape due to long distance, sensor resolution, and partial occlusion leading to an increase in F_1 score.

▶ the estimated velocity and heading angle of objects depend on the motion between the corresponding point clusters with consecutive scans. In most typical driving scenarios for urban environment, the accuracy of velocity and heading angle estimation has been improved.

The GMFA tracks the moving objects and estimates their states. In our approach, compared with previous studies, each point is depended on clustering using Euclidean distance. Since the correspondence between points is determined by the similarity of shape and the distance between the average position of the cluster, the correspondence between points in successive scans can be established even with a small computational load. After establishing the correspondence between the clusters, Iterative Closest Point (ICP) method is applied to perform a match for each cluster, and also the states of the moving

objects are estimated through the Extended Kalman Filter (EKF) based on the movement direction and distance of the mean position of the cluster.

The multiple object state estimation is conducted by using clusters obtained in grouping process. In order to estimate the motion states of the object, EKF is designed by utilizing processed signals as measurements and motion information of the target as output. The motion information of the vehicle was defined as filter input. The state vector and input vector for the filter were defined as follows:

$$x_n = \begin{bmatrix} p_{n,x} & p_{n,y} & \theta_n & v_{n,x} & \gamma_n & a_{n,x} & \dot{\gamma}_n \end{bmatrix}^T \tag{4.13}$$

$$u = \begin{bmatrix} v_x & \gamma \end{bmatrix}^T \tag{4.14}$$

The coordinate systems and the seven states of filter are depicted in Figure 4.25. In this study, we use two coordinate systems: a fixed global coordinate system, $O_g x_g y_g$, and a body fixed coordinate system moving around the rear axle of subject vehicle, $O_e x_e y_e$. The $p_{n,x}$, $p_{n,y}$ stand for the mean position of the cluster, and θ_n indicates the heading angle of the object in respect of $O_e x_e y_e$ coordinate system. The $v_{n,x}$ denotes the vehicle velocity in the direction of $O_g x_g y_g$ frame. The γ_n, $a_{n,x}$, and $\dot{\gamma}_n$ describe the rate of heading angle, the acceleration, and the angular acceleration of subject vehicle

with respect to $O_g x_g y_g$, respectively. The v_x, γ stand for the velocity and the rate of heading angle of subject vehicle at the frame of $O_g x_g y_g$.

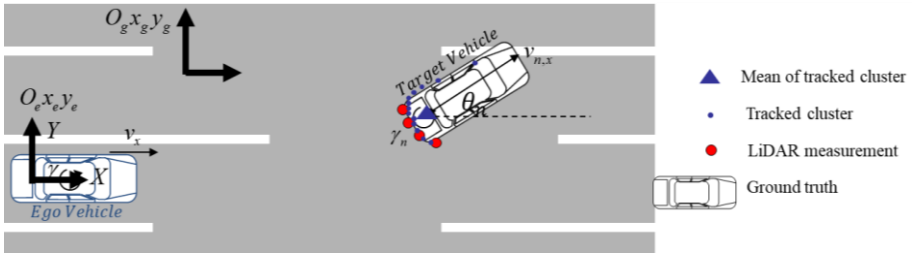


Figure 4.25. States of moving object in GMFA

4.3.1. Prediction of Geometric Model-Free Approach

Each track is predicted through process update of the discrete-time EKF using the model as expressed in equation (4.15).

$$\begin{aligned}\dot{x}_n &= \mathbf{a}\left(x_n, u\right) + \mathbf{w} \\ &= \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 \end{bmatrix}^T + \mathbf{w}\end{aligned} \tag{4.15}$$

where,

$$a_1 = v_{n,x}\cos\theta_n - v_x + p_{n,y} \times \gamma$$
$$a_2 = v_{n,x}\sin\theta_n - p_{n,x} \times \gamma$$
$$a_3 = \gamma_n - \gamma \qquad a_4 = a_{n,x}$$
$$a_5 = \dot{\gamma}_n \qquad a_6 = -k_a \qquad a_7 = -k_{\dot{\gamma}}$$

where $\mathbf{a}(x_n, u)$ is the state function; and $\mathbf{w}$ is the white Gaussian noise with covariance matrix W_k, respectively. Also, the discretization of the process model has been accomplished [Kim,'15b]. The state and the observation functions are linearized, as shown in the following equations:

$$A_{[i,j]} = \frac{\partial \mathbf{a}_{[i]}}{\partial x_{[j]}} \tag{4.16}$$

where A is the Jacobian matrix of the state function. The track index n is omitted for clarity. By neglecting the higher-order terms of the Taylor series, the *priori* state estimate and the covariance matrix can be given as follows:

$$\begin{aligned} x_{k+1}^{-} &= \mathbf{a}\left(x_k^{+}, u_k\right) \\ P_{k+1}^{-} &= A_k P_k A_k^T + W_k \end{aligned} \tag{4.17}$$

where x_{k+1}^{-}, P_{k+1}^{-}, and x_{k+1}^{+} are the *priori* state at time k+1, the *priori* covariance at time k+1, and the *posteriori* state at time k, respectively.

It is essential to transform the previous clusters to present step $O_e x_e y_e$ in accordance with the static assumption in order to initialize the tracks and evaluate the speed of moving objects. The clusters of the past step, $\mathbf{Z}[k-1]$, are transformed into the present step, $\bar{\mathbf{Z}}[k-1]$, employing dead reckoning using velocity and yaw angle rate of subject vehicle under the static assumption.

4.3.2. Track Management

The track management process can be divided into three categories: the cluster assignment in the present step to the predicted tracks from previous result, the initialization of the new tracks using clusters not yet assigned to the predicted tracks, and the termination of the tracks that have yet to be updated over a given period. The cluster assignment to the predicted object track is conducted through technique of Global Nearest Neighbor (GNN). For a comprehensive analysis of the track management in this research, the $\bar{\mathbf{Z}}[k-1]$, $\mathbf{Z}[k]$, and $\{\bar{\mathbf{Z}}_n[k]\}$ are configured as follows. The $\bar{\mathbf{Z}}[k-1]$ is composed of p clusters, $\{\bar{\mathbf{Y}}_1, \bar{\mathbf{Y}}_2, \cdots, \bar{\mathbf{Y}}_i, \cdots, \bar{\mathbf{Y}}_p\}$, and each $\bar{\mathbf{Y}}_i$ consists of n_i two-dimensional points. The $\mathbf{Z}[k]$ consists of q clusters, $\{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_j, \cdots, \mathbf{Y}_q\}$, and $\{\bar{\mathbf{Z}}_n[k]\}$ also comprise N clusters, $\{\bar{\mathbf{Z}}_1, \cdots, \bar{\mathbf{Z}}_n, \cdots, \bar{\mathbf{Z}}_N\}$. The feature vector, f, for each cluster A, in GNN is determined previously as shown in equation (4.18).

$$\begin{aligned} f &\triangleq [x, y, \lambda_{MAX}, \lambda_{\min}]^T \\ [x, y] &= mean(A) \\ [\lambda_{MAX}, \lambda_{\min}] &= eig(\mathrm{cov}(A)) \qquad when\ \lambda_{MAX} \geq \lambda_{\min} \end{aligned} \tag{4.18}$$

The feature vector is a four-dimensional vector containing of a mean point position and eigenvalues of the cluster's covariance matrix. The eigenvalues provide information of feature geometry regardless of rotation. A weighted 2-norm is defined as a Euclidean distance in 4-D feature space. If the distance

between $\bar{\mathbf{Z}}_n$ and $\mathbf{Y}_j$ is less than a predefined threshold, $\mathbf{Y}_j$ indicates a measured value of the *n*-th track $\mathbf{Z}_n$.

Once the assignment of the measurements to the predicted tracks is completed, the track initialization and interruption are performed. If the track is not continuously updated for more than 30% of the duration of life or for consecutive three steps, the track is dropped. The track initialization presents a new track generation by using clusters ($\bar{\mathbf{Y}}_i$, $\mathbf{Y}_j$) that has not yet been assigned to a track. If the distance between $\bar{\mathbf{Y}}_i$ and $\mathbf{Y}_j$ is less than the predefined threshold of distance, a corresponding relationship for creating a new track is established. The position, velocity, and heading angle of object are initialized through ICP based point matching, and the other elements are initialized to zero.

4.3.3. Measurement Update

The EKF structure is used to conduct the measurement update. Since the process model of the object was described in subsection 4.3.1, the measurement and the actual calculation of the assigned cluster for *n*-th track, $\mathbf{Z}_n$, are discussed in this subsection. In this study, the three measurements collected via $\mathbf{Z}_n$ include the x, y directional position and the heading angle of the target objects. When $\mathbf{z}_n$ is a measurement vector of the EKF, $\mathbf{z}_n$ is expressed as a 3D vector as following:

$$\mathbf{z}_n = [h_{n,1}, h_{n,2}, h_{n,3}]^T \tag{4.19}$$

The three elements of $\mathbf{z}_n$ express the mean position of point cloud cluster and yaw angle of the object at $O_e x_e y_e$, respectively. The position of the *n*-th track is considered the average of the matched $\overline{\mathbf{Z}}_n$ after matching $\overline{\mathbf{Z}}_n$ to $\mathbf{Z}_n$ by using ICP. The movement direction of the moving object represents the direction of the displacement vector from the mean position of $\hat{\mathbf{Z}}_n[k-1]$ to the mean position of matching $\overline{\mathbf{Z}}_n$. In Figure 4.26, these measurements are depicted with different colored dots, and the measurement model based on them is linear as derived in (4.20), assuming that the measured values associated with white Gaussian noise with a covariance matrix of $\mathbf{V}_n$.

$$\begin{aligned}
&\mathbf{z}_n[k] = \mathbf{H}_n x_n[k] + \mathbf{v}_n[k] \\
&\mathbf{v}_n[k] \sim N\left(0, \mathbf{V}_n[k]\right) \\
&\mathbf{H}_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{4.20}$$

With this measurement, the state can be updated by calculating Kalman gain. The posteriori estimates can be given by:

$$
\begin{aligned}
K_{k+1} &= P_{k+1}^{-} H_k^T \left(H_k P_{k+1}^{-} H_k^T + R_k \right)^{-1} \\
x_{k+1}^{+} &= x_{k+1}^{-} + K_{k+1} \left(z_{k+1} - H_k x_{k+1}^{-} \right) \\
P_{k+1}^{+} &= \left(I - K_{k+1} H_k \right) P_{k+1}^{-}
\end{aligned}
\tag{4.21}
$$

where K_{k+1}, x_{k+1}^{+} and P_{k+1}^{+} are, respectively, the Kalman gain, the *posteriori* estimate, and the *posteriori* covariance.

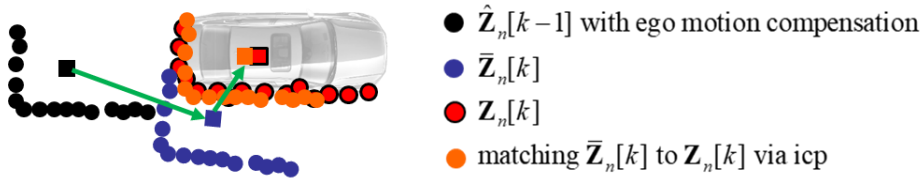


Figure 4.26. The measurement of n-th track from corresponded cluster. The triangle denotes mean point of each cluster.

4.3.4. Performance Evaluation via Vehicle Tests

The proposed object tracking algorithm has been verified by comparing to geometric model-based tracking (MBT) algorithm in respect of object detection and state estimation. The MBT is the representative tracking method among the detect before track approaches. MBT extracts the practically possible feature candidates of objects from the present point cloud signals, and tracks the shape candidates using the multiple hypothesis tracking structure proposed in [Cho,'14]. After clustering in the point clouds of present step, the shape candidates were extracted by using the scheme of bounding box and the virtual ray. The performance of state estimation accuracy was verified by comparing to MBT approach in regard to the standard deviation of the state estimation

error. With the purpose of the error analysis, the driving data including situation of lane keeping and lane change were acquired at various relative positions using the RT-range. The data log of Seoul National University (SNU) Kwanak campus is used to determine the results. The test-driving road included a variety of urban environments such as intersections, crosswalks, and speed bumps. Each data frame was labeled with moving objects detected from a camera located in front windshield, and 540 number of moving objects were labeled including vehicles, buses, motorcycles and so forth.

The results of target detection are shown in Table 10. As explained in Table 10, the F_1 score for the proposed approach was around 59% higher than that for MBT approach since both precision and recall had increased. The precision and recall were improved by 0.062 and 0.478 respectively. Increased precision indicates a decrease in the number of false alarms, and significantly improvement of recall indicates the reduction in the frequency of false negative results.

Table 10. Comparison result of GMFA and MBT based on data log of SNU Kwanak Campus.

Method	Moving Objects	Detected Objects	Correctly Detected	Precision	Recall	F_1score
GMFA	540	568	486	0.856	0.900	0.877
MBT	540	287	228	0.794	0.422	0.551

As the performance of tracking was obtained without assumption of the moving objects' shape, the efficient detection and tracking of diverse moving

objects faced in the urban situation were successfully performed. The speed and heading angle of the vehicle were estimated with a high degree of confidence through points correlation in consecutive scans.

Chapter 5 Computational Load Management

The automated driving system consists of perception, positioning, decision-making, and control, and the real-time stability of the system must be secured for safe driving. For the whole system to operate normally, a designated calculation cycle must be ensured based on the evaluation of hardware capabilities and algorithm performance. The computational load of the environment perception algorithm is very high in terms of real-time stability since the computational load of the perception module that processes environment information takes up a large portion of all resources. Therefore, this study focuses on lidar-based environmental recognition technology and proposes a computational load management strategy to ensure real-time reliability of environment recognition systems for automated driving.

The adaptive ROI-based environment perception algorithm designed in Chapter 3 and 4 has improved efficiency, scalability, and accuracy compared to the existing algorithm. Although this improves efficiency, there is still a possibility that the operation fails within a given period, so this design cannot guarantee that real-time stability is achieved. To improve the real-time stability of the target algorithm, it is necessary first to identify and efficiently use the resources allocated to the cognitive algorithm in the target system. In terms of practical application, the computational load is regarded as the computation time. The lower processing module constituting the environment-aware

algorithm must complete the operation within the allocated period for smooth operation. Data that senses the surrounding environment using lidar correlates with the amount of data and the point cloud data depending on the road structure and traffic conditions, and the deviation can be quite significant. Unlike most algorithms with fixed input and output sizes, lidar-based environment recognition algorithms have variable input and output data sizes. In order to reflect these data characteristics and the characteristics of the entire system, a computational load management strategy is established based on the data analysis of the target automated driving system implemented on the vehicle.

The structure of the automated driving system developed in this laboratory is shown in Figure 5.1. Lidar-based algorithms applied in this system are mounted along with motion planning on PC devices shown in shades of blue in Figure 5.1.

To determine sampling cycle of system, various factors should be considered: hardware capability and requirement of system performance. The automated driving system required to be able to understand the real-time traffic environment and react to it fast enough. However, the actual performance requirement for automated driving system is still largely undefined. According to previous work in ADAS, the reaction time of ADS is determined by two factors: frame rate and processing latency. The frame rate determines how fast the sensor data can be fed into the process engine, and the processing latency of perceiving scenes and making operational decisions determines how fast the system can react to the acquired sensor information. Human drivers take varying amount of time to respond based on the level of expectation and action

chosen. The fastest possible action by a human driver takes 100-150ms [Newell,'85, Thorpe,'96]. To secure better vehicle safety, automated driving systems must be able to react faster than human drivers, which recommends the processing latency should be within 100 milliseconds [Lin,'18]. To react quickly to the constantly changing traffic condition, the system should be able to react faster than human reaction time, which suggests a frequency of once every 100 milliseconds. This processing frequency aligns with the industry standards of Mobileye [Shalev-Shwartz,'16] and the design specifications of Udacity [Udacity,'17]. In consideration of the system performance constraint determined above, target hardware specification and the sensor's performance, the sampling cycle is designed at 10Hz.

Due to the interlocking characteristics of the LabVIEW and MATLAB software applied, environment awareness and motion planning are designed to perform serial operations in our ADS. Since the sum of the execution time of two modules on the same hardware should not exceed a predetermined period, the resource allocated to the environment recognition algorithm is calculated using the relatively small variation in the computational load of the motion planning module. By applying multiple linear regression to the driving data of the target system, the computational load of the main functions of the perception algorithm is estimated before the algorithm is executed. In order to prevent execution time from being exceeded based on the expected processing time, processing time reduction is performed in which the data used for processing is reduced sequentially based on ROI. Nevertheless, when the actual execution time of the algorithm continuously exceeds the allocation criteria, the

driving control ability is partially limited to ensure driving safety and real-time stability of the proposed system.

The performance of the proposed algorithm is verified through data-based simulation with driving logs. It is shown that the perception module's computation efficiency performance can be significantly enhanced.

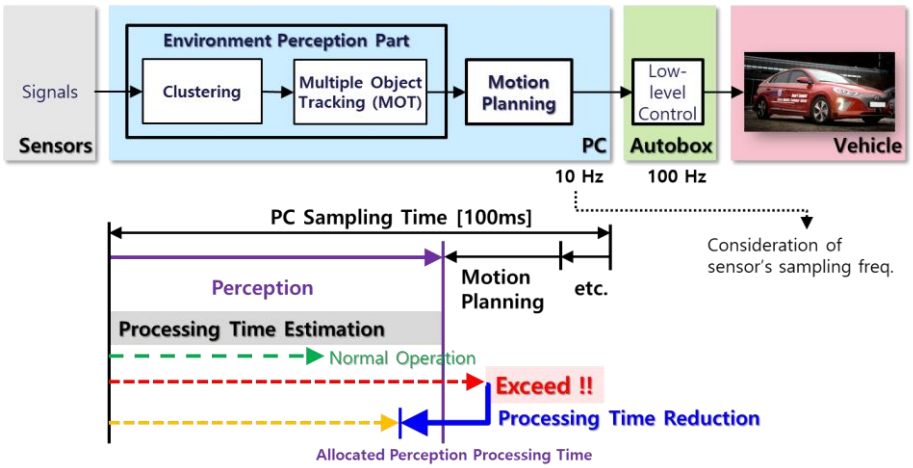


Figure 5.1. Scheme of processing time estimation and reduction.

5.1. Processing Time Analysis of Driving Data

In order to ensure the real-time stability of the proposed algorithm under given conditions, it is necessary to identify the allocated resources and effectively use them accurately. The ability of the algorithm to keep a proper execution cycle under given execution conditions is a criterion for evaluating the algorithm's real-time computational stability. For the real-time performance of the algorithm proposed in this study, we must first identify the resources allocated to the algorithm.

The main modules of the automated driving system we are developing are composed of PC and Autobox, as shown in Figure 5.1. Among them, algorithms such as environment perception and motion planning are installed on PC. Since the resources are shared among the algorithms of the same device, resources must be allocated individually for each algorithm. As mentioned above, in the recognition algorithm using the environment sensor information, the dimension of input and output data is variable, and the deviation is significant. On the other hand, the calculation time required for motion planning and other algorithms is relatively small because the variation of operation time by cycle is relatively small. By performing statistical analysis on the actual driving data of the target system, the processing time required for algorithms other than recognition can be obtained. Using this, it calculates the resources that can be used by the proposed environmental awareness algorithm while guaranteeing the performance of other algorithms in the PC environment.

Motion planning is the most computationally loaded algorithm in PC, excluding the environment recognition algorithm. Therefore, to obtain the maximum resources available in perception computation, an analysis of the motion planning algorithm should be preceded first. Motion planning is performed using all information, including environmental recognition results. The motion planning of the automated driving system applied in this study is designed based on MPC. The primary process consists of four steps, as shown in Figure 5.2. The safety envelope is determined using the environmental cognition results, maps, and localization results, and the ego vehicle model for the MPC is determined. Motion optimization is finally performed through mode decision on lane keeping and lane change.

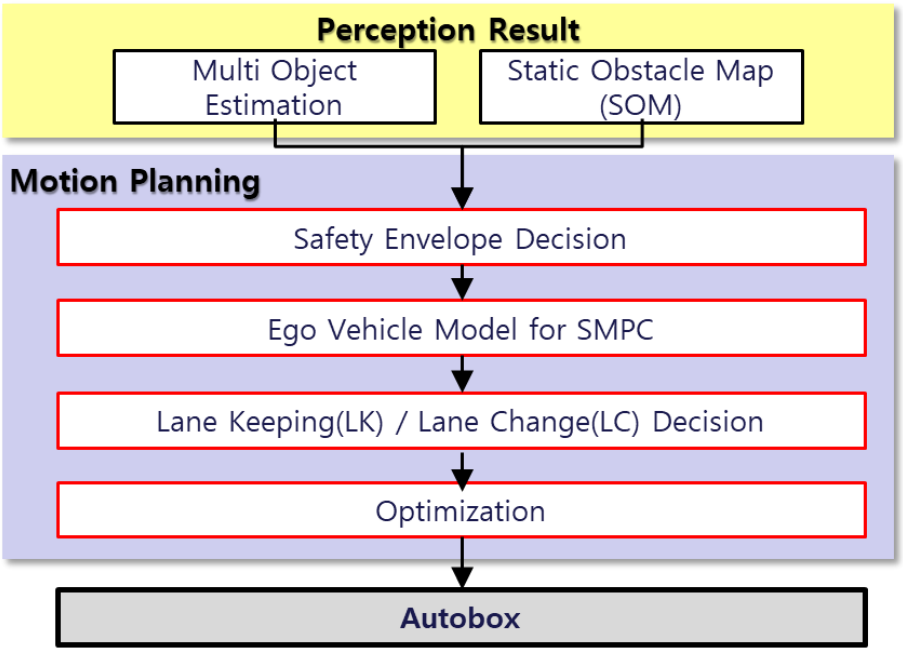
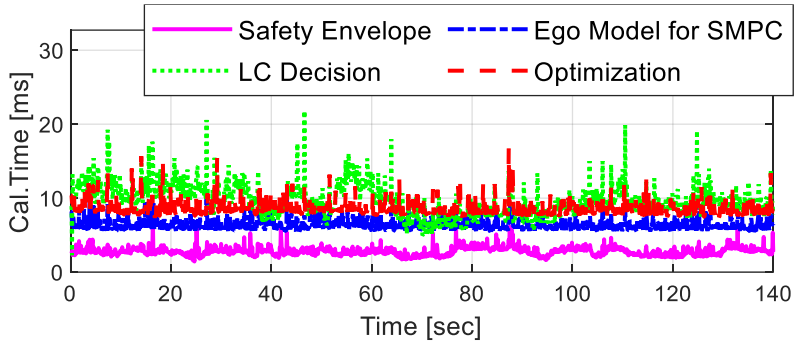


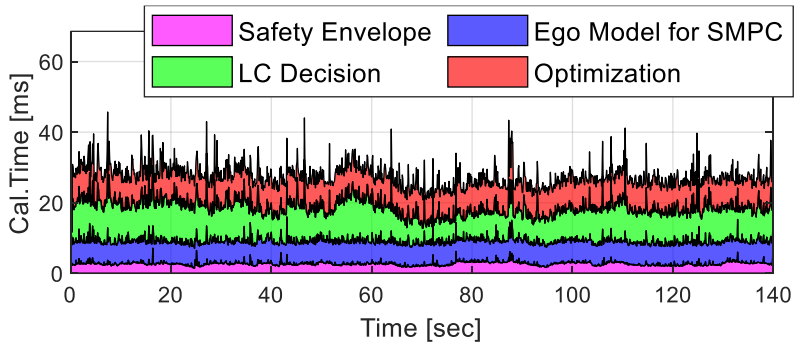
Figure 5.2. Algorithm flow of motion planning after environment perception.

Using the actual driving data of the subject automated vehicle, the computation time of the functions that make up the motion planning depicted in Figure 5.2 are analyzed. Figure 5.3 shows the result of analyzing the calculation time of MPC-based motion planning functions using driving data. Figure 5.3 (a) shows the computation time for each configuration function. In other functions except for LK/LC decision, it is possible to see that the operation time is constant regardless of the situation, considering intermittent noise. On the other hand, the LK/LC decision function can see some variation in computations depending on the driving situation, but the deviation is not significant. Figure 5.3 (b) shows the processing time of motion planning over time by accumulating the result of Figure 5.3 (a). Except for some occasional noise, it can be seen that the total computing time and proportion are constant. Figure 5.3 (c) shows the histogram of the time required for each function. Many functions have small standard deviations, but as mentioned earlier, LK/LC decision shows significant variations in computational time depending on the situation. The algorithms of this automated driving system on PC are designed in LabVIEW and MATLAB software in the Windows OS environment. Since the real-time performance guarantee is not an optimal development environment, the intermittent noises showed in Figure 5.3 (a) and (b) plots are presumed to be due to such limitations. Figure 5.3 (d) shows the execution time in overall motion planning. The green dashed line represents the total computation time set in the PC. The dashed blue line represents the maximum computation time, and the red dashed line represents the minimum computation time. In view of reflecting all these characteristics in order to secure real-time

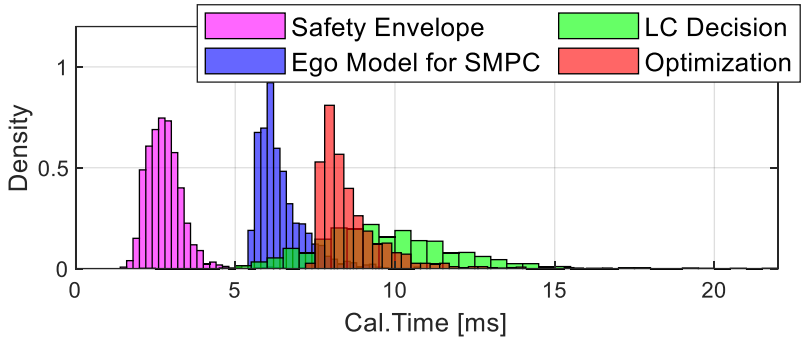
stability in the target environment, the maximum time is determined as the acquisition time for the smooth execution of motion planning.



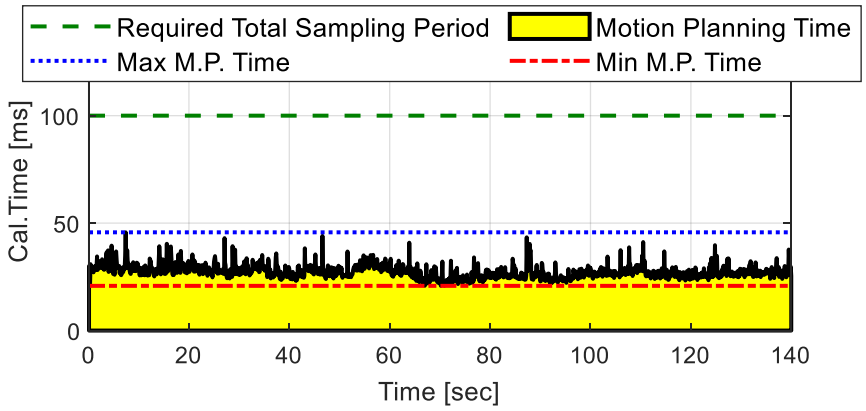
(a) Processing time by each motion planning function



(b) Stacked processing time by each motion planning function



(c) Histogram of motion planning function time

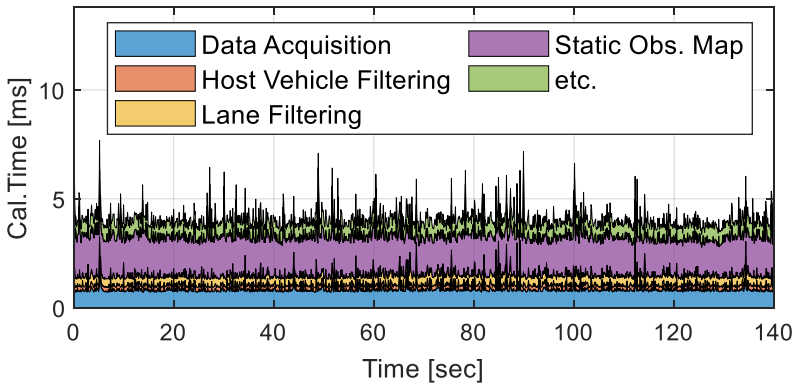


(d) Total motion planning processing time

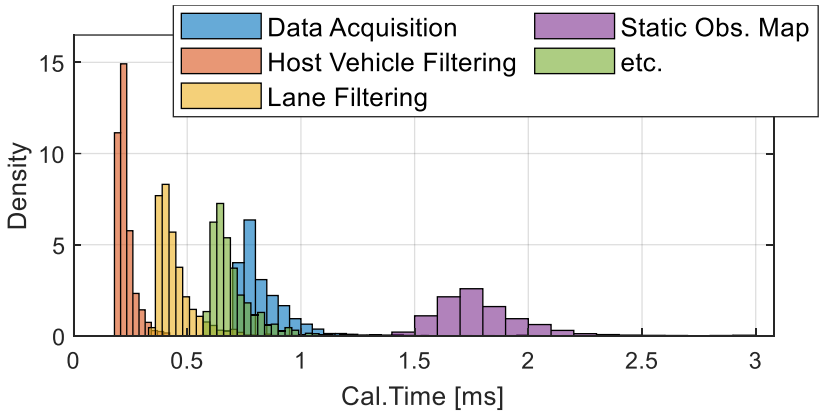
Figure 5.3. Processing time analysis of motion planning based on driving data.

Data analysis has been conducted on the processing time of the functions required to perform the remaining functions in the PC device. The computation time of the remaining functions was analyzed to reflect the actual system characteristics, although the computations are relatively small compared to the motion planning and perception algorithm. Figure 5.4 (a) is a cumulative plot of the computation time, and the shaded color expresses the ratio of computation time for each function. Figure 5.4 (b) shows the histogram of the time required for each function. In most cases, it takes about 1 or 2 milliseconds. The processing time for the motion planning and other functions analyzed earlier, based on the total allocated time, is depicted in Figure 5.5. The area filled in yellow is the time required for motion planning, and the area filled in cyan is other processing time, which is relatively small. As mentioned above, to take into consideration the performance characteristics of the H/W and S/W intermittently, the decision is made based on the worst case. Thus, as shown by

the red dashed line in Figure 5.5, 50.29 milliseconds of the total time of 100 milliseconds can be allocated to an algorithm that is lidar-based.



(a) Stacked Processing time by other major function



(b) Histogram of each motion planning function

Figure 5.4. Processing time analysis of other function except motion planning based on driving data.

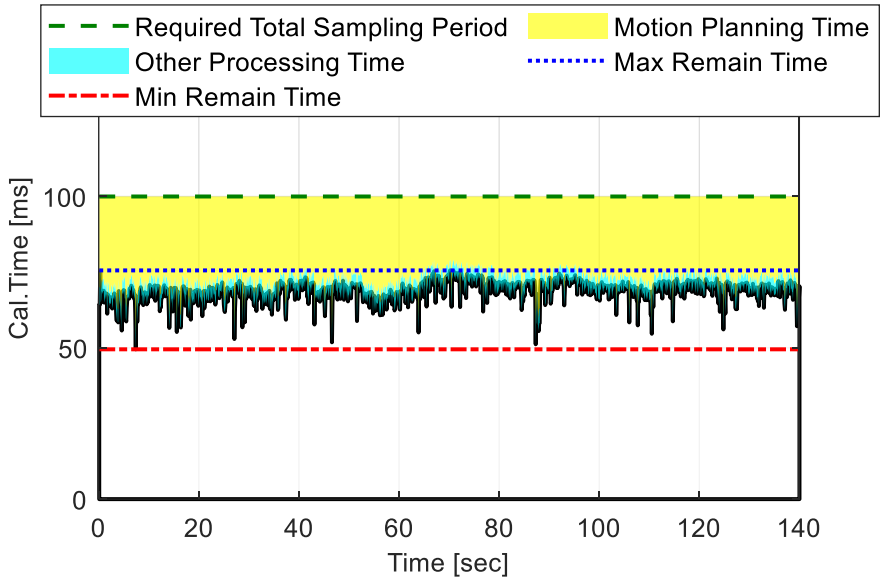


Figure 5.5. Evaluation result of remained processing time for perception.

5.2. Processing Time Estimation based on Multiple Linear Regression

From the results in Section 5.1, the resources allocated to lidar-based cognitive algorithms in the target system were identified in terms of computation time. The algorithm should be designed to perform operations within allocated resource conditions to increase the real-time reliability of the target algorithm. The adaptive ROI-based environmental cognition algorithm designed in Chapter 3 has improved efficiency, scalability, and accuracy compared to the existing algorithm. Although the efficiency is improved, there is still a possibility that the operation fails within a given period. Therefore, it is necessary to reduce processing load to ensure real-time stability. To manage computational load of the system, this section proposes computational time prediction techniques in functional units that constitute the algorithm.

Through data analysis, the lidar-based perception algorithm of the automated driving system correlates with the characteristics of input/output data, processing results of the previous cycle, and processing time of the current cycle. A processing time estimation model is constructed for clustering and multi-object tracking, which are functions that are highly resource-intensive to secure real-time computational reliability. As shown in Figure 5.6, the factors that account for a large proportion of the computation of the two functions were selected among the variables in the perception algorithm. By fitting the various regression models, an appropriate processing time prediction model was

determined. Multiple linear regression models were selected to reflect the algorithm and execution environment of the target system by applying them to many driving data as well as statistical fitting results.

The strategies for estimating computation time prior to actual processing using the processing time prediction model obtained in this section to determine whether execution can be performed normally with the allocated resources and preventing failures will be covered in the next section.

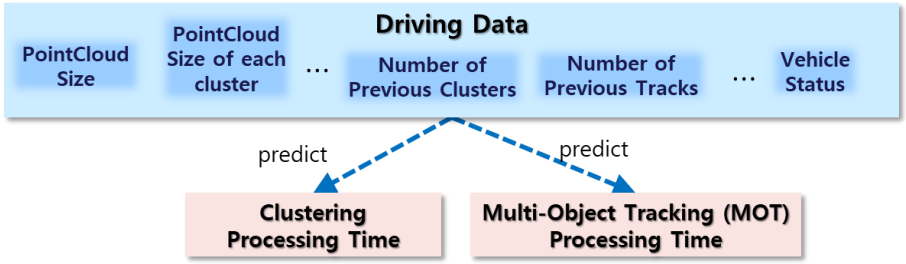


Figure 5.6. Scheme of processing time estimation based on multiple linear regression.

5.2.1. Clustering Processing Time Estimation

In this subsection, we predict the processing time of the clustering process that requires considerable computational power among the lidar-based cognitive algorithms applied to autonomous driving systems. By analyzing the actual driving data of the target autonomous driving system, we extract the main factors affecting the clustering process and determine the relationship by applying the linear regression technique. By predicting the processing time based on only the information of the main factor before operating, it is expected

that it is possible to determine in advance whether a successful operation can be performed with the resources allocated to the lidar-based perception module.

The clustering refers to a process of grouping point cloud data into adjacent group units by considering scattered point cloud data such as Euclidean distance. In subsection 3.2.3, clustering processing was described in detail. For the distributed point cloud to be classified as a group, the point-to-point Euclidean distance must be smaller than the distance set as the threshold. In the process of clustering in this way, the distance between each point must be calculated one by one. An intuitive method is to perform sequential iterations of points in the entered point cloud, where a point is determined, and the distance to the remaining points is compared based on that point. Another method is to calculate the distance between all points to be processed in a vast matrix form and extract the point-to-point relationship through the processing process. In both methods, it can be determined that the number of operations and the dimension of the matrix are determined according to the size of the input data. Therefore, clustering processing has a performance characteristic that depends on the dimension of the input point cloud under the condition that hardware specifications and software characteristics are similar. This relationship is confirmed by the data distribution between the input point cloud size and the clustering processing time, as shown in Figure 5.8 (b).

As mentioned above, it can be seen that the input point cloud size is intuitively proportional to the clustering processing. However, to construct a processing time prediction model with only one factor, the margin of error due to the distribution of the point cloud can be quite large. To find out the factors

related to the clustering operation, we analyzed the relationship between the parameters in the algorithm and various signals and processing time. Road facilities, such as walls, guardrails, and buildings, do not move, and treating them as moving objects reduces not only recognition accuracy but also increases the computational load. Boundary information of clusters exceeding a specific vehicle size is applied to construct a static obstacle map, and it is classified as an object that cannot move in the clustering process and is not used for MOT. However, as the cluster size increases during the clustering process, the correlation between already clustered points and the remaining points needs to be checked, and thus, it is estimated that a considerable computational load occurs. Therefore, the majority of data that is not determined to be a valid cluster moving in the clustering results are named as rejected point cloud because they are dropped from the cluster because they are huge objects causing computational load. Figure 5.8 (c) shows the relationship between the cluster output rejected point cloud and the clustering time. The relationship between the clustering processing time and the two main factors has a constant tendency, as shown in Figure 5.7.

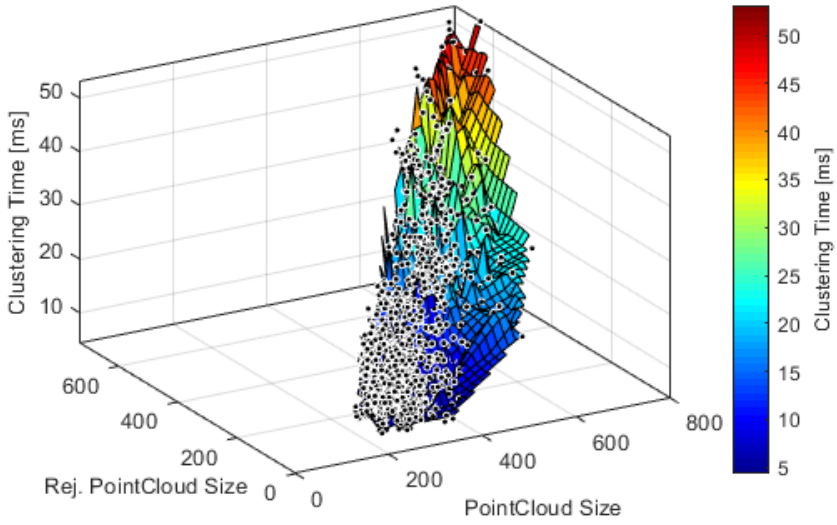
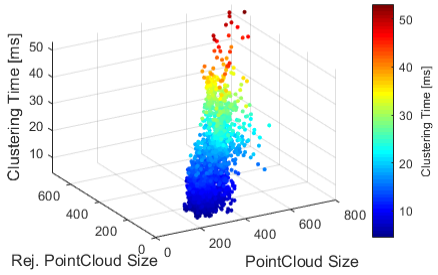
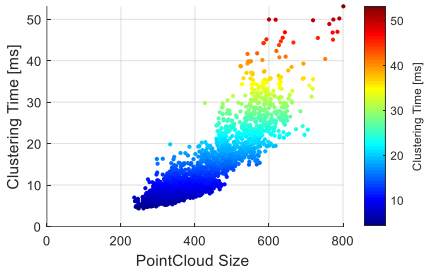


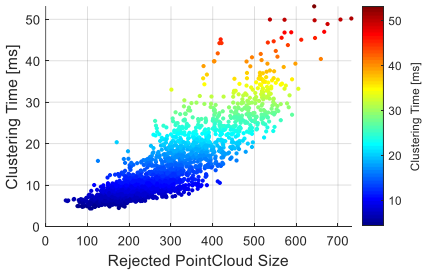
Figure 5.7. 3-D scatter plot of clustering time with P.C. size and rej. P.C. size.



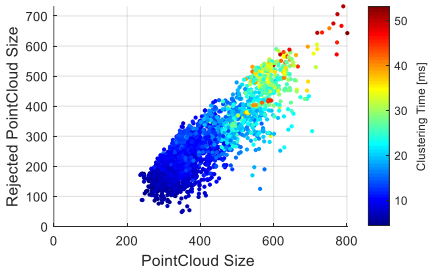
(a) 3-D Scatter plot of clustering time



(b) P.C. size – Clustering time (XZ)



(c) Rej. P.C. size – Clustering time (YZ)



(d) P.C. size – Rejected P.C. size (XY)

Figure 5.8. Scatter plots of clustering time affecting factors.

Two major factors were selected from several factors related to clustering computational complexity: point cloud size and rejected point cloud size. Since the update frequency of the environmental information acquired by the lidar sensor is generally about 10 to 25 Hz, the scan period is short. Therefore, it can be reasonably assumed that a similar signal distribution exists between the scans without a large environmental change. In this way, the rejected point cloud size of the previous cycle can be applied to the current cycle because the results processed in the previous cycle have a distribution that is quite similar to the results of the current cycle. We apply these two factors to multiple linear regression to determine the clustering time prediction model.

To determine the most reasonable model using the lidar environmental signal log stored in driving data, it is necessary to combine the data and the appropriate model. Table 11 shows the characteristics of driving data to perform multi-linear regression.

Table 11 Data description for multiple linear regression of clustering.

Variable	Min.	Max.	Median	Mean	Standard deviation
Clustering Time [ms]	4.3489	53.1293	9.3084	12.3478	7.5718
Point Cloud Size	236	801	355	391.7945	100.0276
Rejected Point Cloud Size	48	732	225	257.8151	106.8280

A reasonable regression model form is required for proper fitting. Referring to the data distribution in Figure 5.8, four models are selected as candidates as shown in (5.1).

$$
\begin{aligned}
&Model\ C1: T_{cluster} \sim \beta_0 + \beta_1 N_{Point Cloud} \\
&Model\ C2: T_{cluster} \sim \beta_0 + \beta_1 N_{Point Cloud,rej} \\
&Model\ C3: T_{cluster} \sim \beta_0 + \beta_1 N_{Point Cloud} + \beta_2 N_{Point Cloud,rej} \\
&Model\ C4: T_{cluster} \sim \beta_0 + \beta_1 N_{Point Cloud} + \beta_2 N_{Point Cloud,rej} + \beta_3 N_{Point Cloud} N_{Point Cloud,rej}
\end{aligned}
\tag{5.1}
$$

Models C1 and C2 are simple linear regression models for each factor. C3 is a multiple linear regression model consisting of the sum of two main factors. Fitting was performed on the data described in Table 11 and the four regression models in Equation (5.1). The results are shown in Table 12.

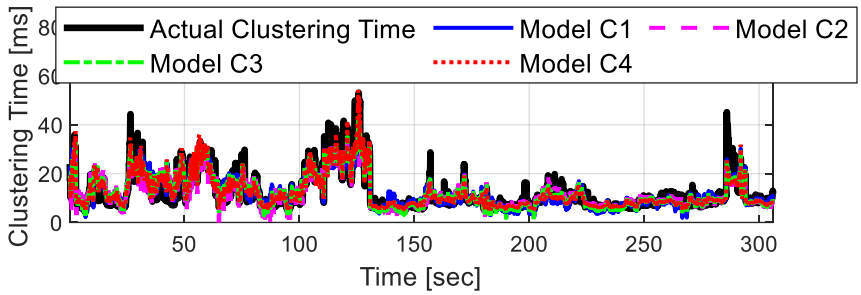
Table 12 Multiple Linear Regression Result of Clustering Time Estimation by models.

Model	RMSE	R^2	Adjusted R^2	F-statistic vs. constant model
C1	3.46	0.792	0.792	1.16e+04
C2	3.52	0.783	0.783	1.11e+04
C3	3.08	0.835	0.835	7.73e+03
C4	2.88	0.856	0.856	6.05e+03

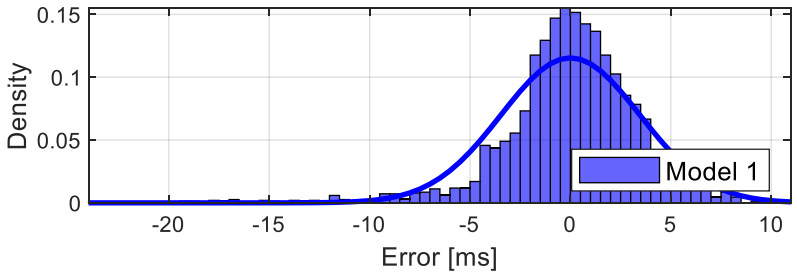
To select the most reasonable model among the four regression models, we applied the data to the target data and compared the output with the reference data and evaluated the difference. Root Mean Square Error (RMSE) represents

the sum of the squares of the residuals and the least error of the C3 and C4 models. R^2 is the coefficient of multiple determination, and since the model represents the ratio representing the data, it can be said that the larger the data is reflected.

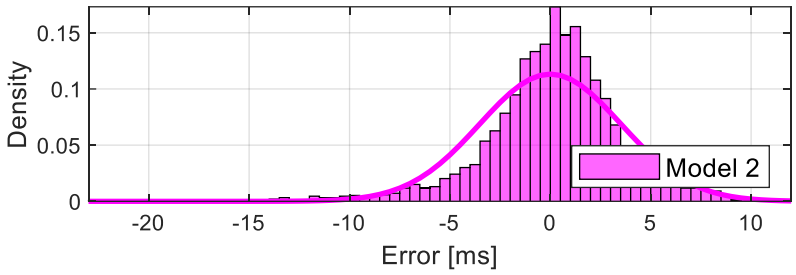
Figure 5.9 illustrates the fitting results of models C1 through C4. Figure 5.10(a) shows a plot comparing the results of the clustering time by the model with the actual data (black-colored solid line) over time. Figure 5.9(a) shows intuitively that multiple linear regression models C3, C4, rather than C1, C2, which are simple linear regression models that only consider the single receiver variable, fit the actual data well. Figure 5.9 (b) through (e) shows a model-specific Residual histogram, where, as in the RMSE shown in Table 12, the response of Model C4 shows better results than other models.



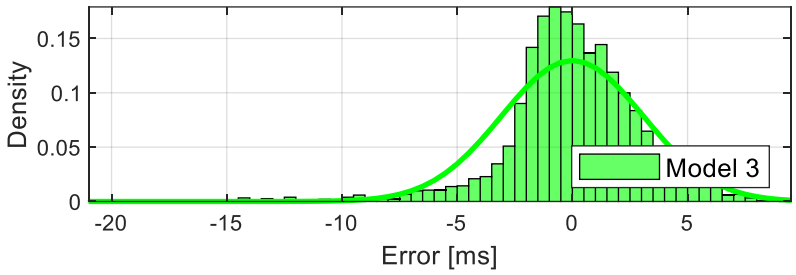
(a) Clustering Time Estimation Result (C1~C4)



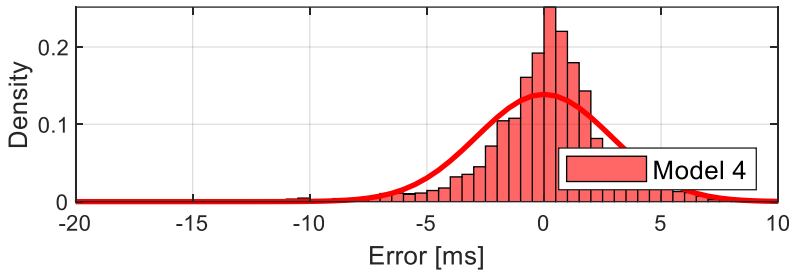
(b) Density distribution of the residual error of model C1



(c) Density distribution of the residual error of model C2



(d) Density distribution of the residual error of model C3



(e) Density distribution of the residual error of model C4

Figure 5.9. Multiple linear regression result of four clustering time models (C1-C4).

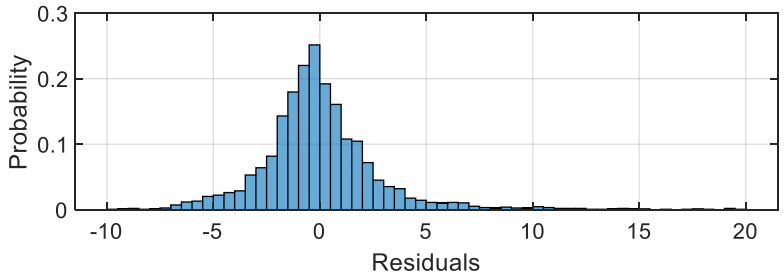
Based on the above results, we determined that the C4 model is the best among the four models for selecting the clustering time estimation model.

$$C4: T_{cluster} \sim \beta_0 + \beta_1 N_{PointCloud} + \beta_2 N_{PointCloud,rej} + \beta_3 N_{PointCloud} N_{PointCloud,rej} \quad (5.2)$$

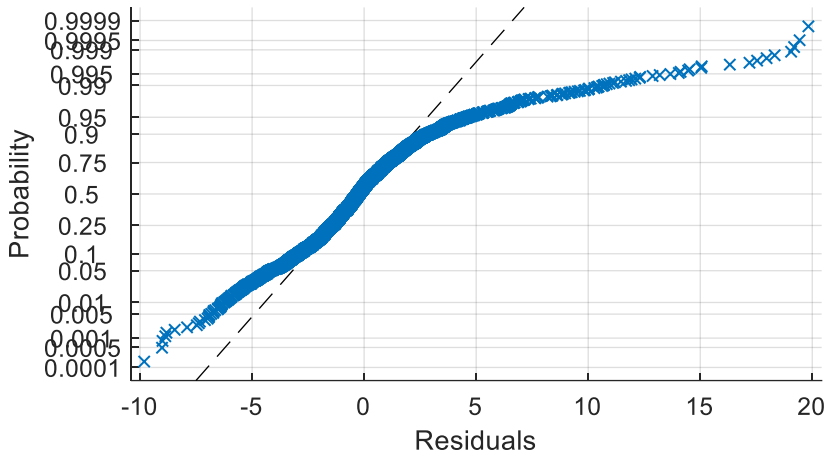
Table 13 shows the analysis of the regression of the selected model C4. Figure 5.10 consists of the histogram of the residual and the normal probability plot of C4. The selected model is used in the computational load assessment to ensure the real-time stability of the algorithm.

Table 13 Properties of selected clustering time prediction model based on multiple linear regression.

	Estimate(β_i)	**SE**	**tStat**	**p-Value**
(Intercept)	1.0211	0.59458	1.7173	0.0086028
$N_{PointCloud}$	0.010224	0.0017011	6.0104	2.0699e-09
$N_{PointCloud,rejected}$	-0.011104	0.0022955	-4.8374	1.3807e-06
$N_{PointCloud}$: $N_{PointCloud,rejected}$	9.2179e-05	4.3736e-06	21.076	3.5058e-92



(a) Histogram of Residuals



(b) Normal Probability Plot of Residuals

Figure 5.10. Residual plots of Selected Model C4.

5.2.2. Multi Object Tracking (MOT) Processing Time Estimation

In this subsection, we determine a model that estimates the computation time for the process of performing multiple object tracking (MOT) with clustering results in the recognition algorithm. Similar to the previous subsection, we extract the main factors that affect the operation of the MOT and apply the driving data-based linear regression technique. The practical MOT computation time can be predicted from an integration perspective by reflecting the hardware specifications and software characteristics because the actual target system data is used. Similarly, since the processing time is predicted using only the information input to the MOT algorithm, it is expected to be effectively used to determine and cope in advance whether normal execution is possible within the allocated computation time.

The MOT of the lidar-based cognitive algorithm proposed in this study is described in detail in previous Section 4.3. GMFA-based MOT performs EKF-based object state estimation using the ICP matching result of point cloud constituting a cluster. ICP matching and individual EKFs are applied as many as the number of tracks created by the MOT structure. Therefore, the computational complexity is increased in proportion to the number of object tracks created in the previous operation cycle. These relationships are shown in Figure 5.12 (c). Besides, since the cluster generated by the clustering process is used to estimate the status of objects moving in the environment, the association process with the existing object tracks is performed as many clusters input to the MOT. For this reason, the number of input clusters can also be a significant factor in the MOT operation time. The relationship between the number of clusters and the MOT operation time is shown in Figure 5.12 (b).

Since the ICP algorithm estimates the state information of objects through ICP matching of the point cloud, it is expected that the number of point groups included in each cluster can be a regressor variable. However, no real correlation with MOT calculation complexity was found through analyzing the actual data. As noted in previous subsection 5.2.1, clusters extracted for application to MOT are already limited in the point cloud count for each cluster input into the MOT process, since classification is preceded by size considerations in the clustering process. For this reason, the point cloud size for each cluster could not be adopted as a regressor variable. Therefore, the regressor variable to obtain the estimation model of MOT computation time was determined by the number of clusters and the number of tracks.

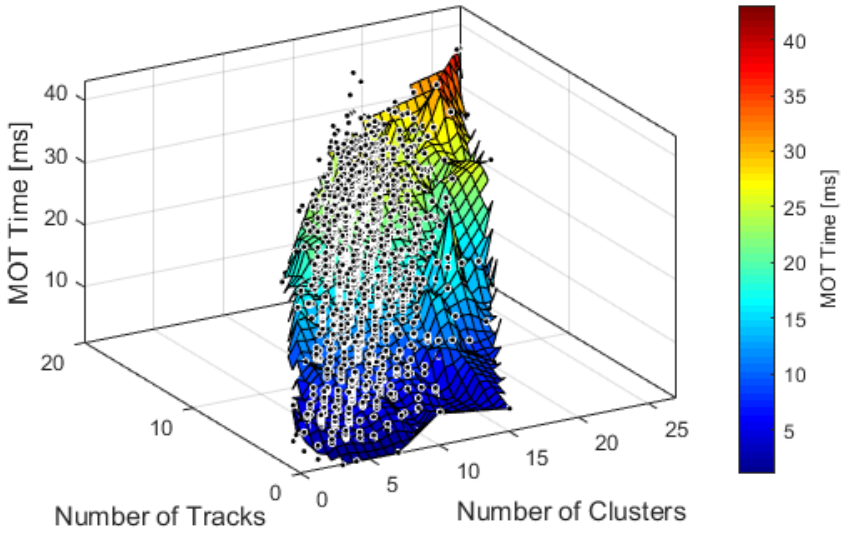
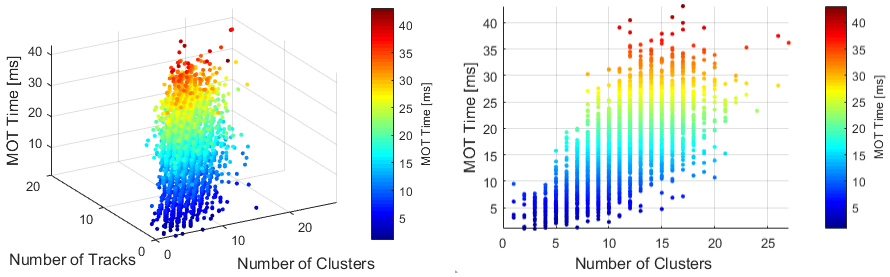
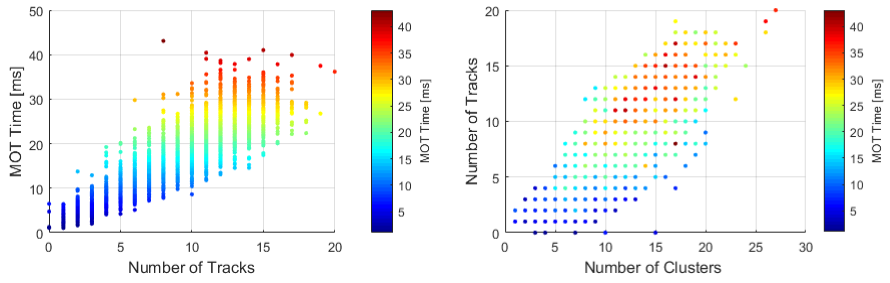


Figure 5.11. 3-D scatter plot of MOT time with point cloud size and rejected point cloud size.



(a) 3-D Scatter plot of MOT time (b) Number of clusters – MOT time



(c) Number of tracks – MOT time (d) Number of clusters – Number of tracks

Figure 5.12. Scatter plots of MOT time affecting factors.

Among the many factors related to MOT computational complexity, two main factors were chosen: the number of clusters and the number of tracks. Therefore, we apply these two factors to multiple linear regression to determine the MOT time prediction model.

To select the most rational model using the driving data, the data and the appropriate model combination candidates are needed. Table 14 shows the characteristics of regressor variables related to the MOT operation of driving data for performing multi-linear regression.

Table 14 Data description for multiple linear regression of MOT.

Variable	Min.	Max.	Median	Mean	Standard deviation
MOT Time [ms]	1.0400	43.0951	17.4848	17.2600	73.4342
Number of Clusters	1	27	12	11.5132	3.7018
Number of Tracks	0	20	9	9.0287	3.7131

A reasonable regression model form is required for proper fitting. Referring to the data distribution in Figure 5.12, four models are selected as candidates as shown in (5.3).

$$
\begin{aligned}
&Model\ M1: T_{cluster} \sim \beta_0 + \beta_1 N_{Cluster} \\
&Model\ M2: T_{cluster} \sim \beta_0 + \beta_1 N_{Track} \\
&Model\ M3: T_{cluster} \sim \beta_0 + \beta_1 N_{Cluster} + \beta_2 N_{Track} \\
&Model\ M4: T_{cluster} \sim \beta_0 + \beta_1 N_{Cluster} + \beta_2 N_{Track} + \beta_3 N_{Cluster} N_{Track}
\end{aligned} \tag{5.3}
$$

Models M1 and M2 are simple linear regression models for each factor. M3 is a multiple linear regression model consisting of the sum of two main factors. In the track association process of the MOT, the multiplication term has a practical meaning because the correlation between the existing tracks and the input clusters is determined through the iterative operation. The fitting was performed on the data described in Table 14 with the four regression models in (5.3), and the results are shown in Table 15.

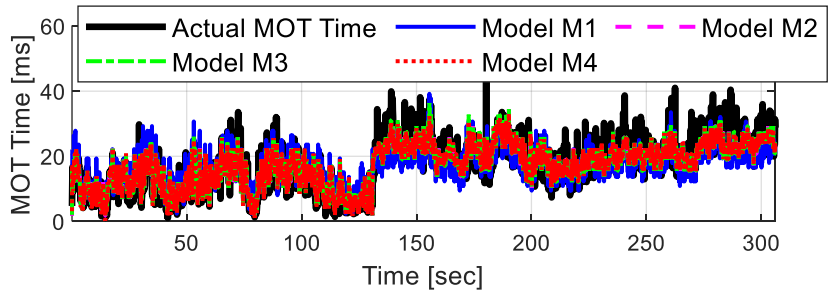
Table 15 Multiple linear regression result of MOT time estimation by models.

Model	RMSE	R^2	Adjusted R^2	F-statistic vs. constant model
M1	5.27	0.498	0.498	3.03e+03
M2	3.78	0.741	0.741	8.76d+03
M3	3.78	0.741	0.741	4.38e+03
M4	3.76	0.744	0.744	2.96e+03

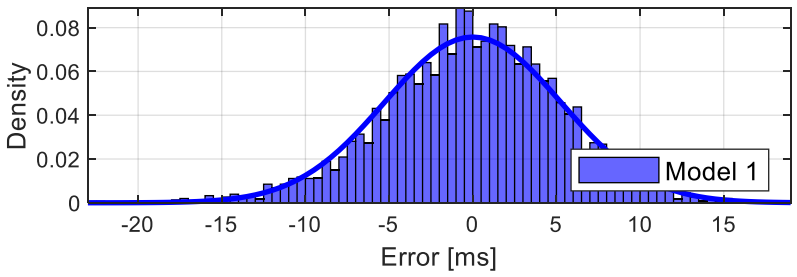
To select the most reasonable model from the fitting results in Table 15, the outputs obtained by applying the models to the target data were compared with the actual reference. RMSE represents the sum of the squares of the residuals, and the M1 model is the largest. The coefficient of multiple determination, R2, represents the ratio at which the model represents the data, so the larger it is, the better the data is reflected. R2 shows that M2, M3, and M4 are models that reflect about 74.1% of data.

Figure 5.14 depicts the fitting results for models M1 through M4. Figure

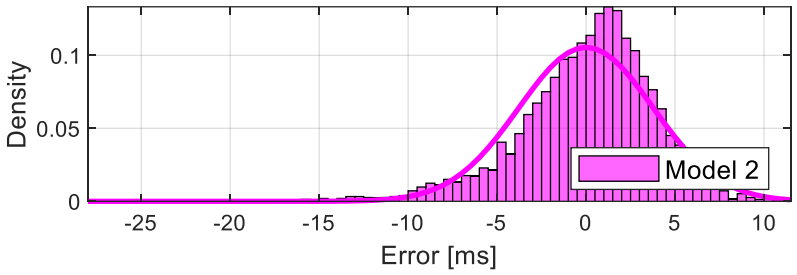
5.14(a) is a plot comparing MOT time estimation results by the model with actual data. In the simple linear regression model considering only the single regressor variable, M1 did not fit adequately compared to M2. It indicates that the computational complexity is highly dependent on the number of tracks due to the structure of the MOT algorithm. In addition, it can be seen intuitively in Figure 5.14 (a) that M2 and M3 and M4, which are multiple linear regression models, are better suited to actual data than M1. Figure 5.14 (b) through (e) shows the residual histogram for each model, where the residuals of models M2, M3, and M4 show better results than the M1 model, as shown in RMSE of Table 16.



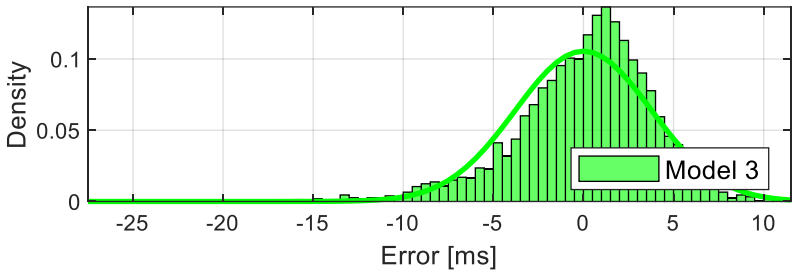
(a) MOT Time Estimation Result (M1~M4)



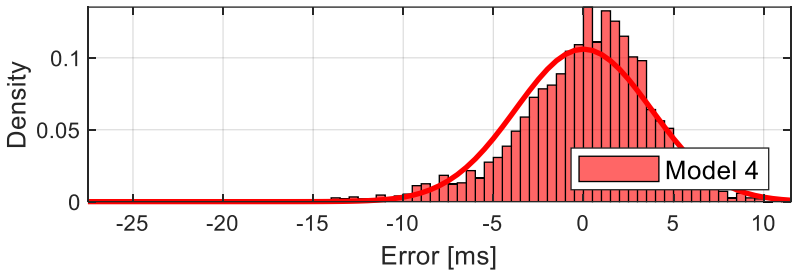
(b) Density distribution of the residual error of model M1



(c) Density distribution of the residual error of model M2



(d) Density distribution of the residual error of model M3



(e) Density distribution of the residual error of model M4

Figure 5.13. Multiple linear regression result of four MOT time models (M1-M4).

From the above results, the fitting performance evaluation of M2, M3, and M4 except for M1 has been similarly derived. When the three models are applied to data not used for fitting, the error of the M4 model is calculated to be the smallest. It is because the regression model of M4 is appropriately

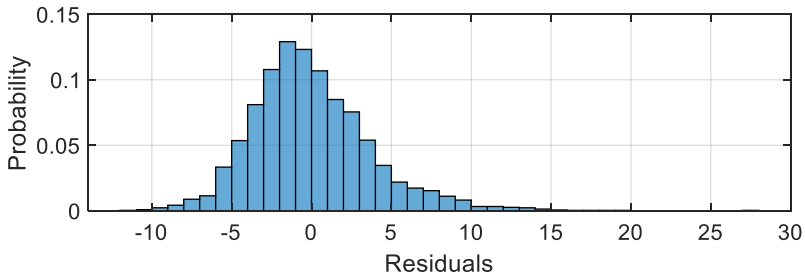
composed of the items reflecting the characteristics related to the MOT calculation load. Therefore, it is determined that the M4 model is most suitable as the MOT time estimation model.

$$M4: T_{cluster} \sim \beta_0 + \beta_1 N_{Cluster} + \beta_2 N_{Track} + \beta_3 N_{Cluster} N_{Track} \tag{5.4}$$

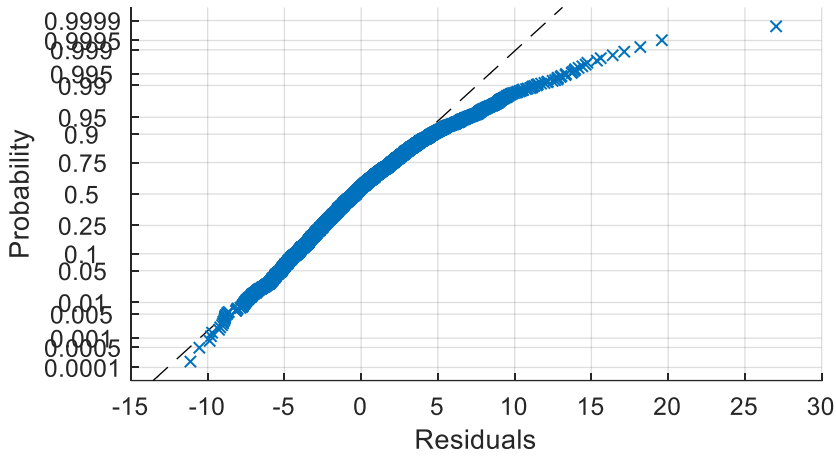
Table 16 shows the analysis of the regression of the selected model M4 and Figure 5.14 consists of the histogram of the residual and the normal probability plot of M4. The selected model is applied to the computational load assessment to ensure the real-time stability of the algorithm.

Table 16 Properties of selected MOT time model based on multiple linear regression.

	Estimate(β_i)	**SE**	**tStat**	**p-Value**
(Intercept)	-0.49918	0.42539	-1.1735	0.02407
$N_{Cluster}$	0.2419	0.048676	4.9696	7.0771e-07
N_{Track}	1.9656	0.056395	34.854	2.0655e-224
$N_{Cluster}:N_{Track}$	-0.024085	0.0042215	-5.7054	1.2719e-08



(a) Histogram of Residuals



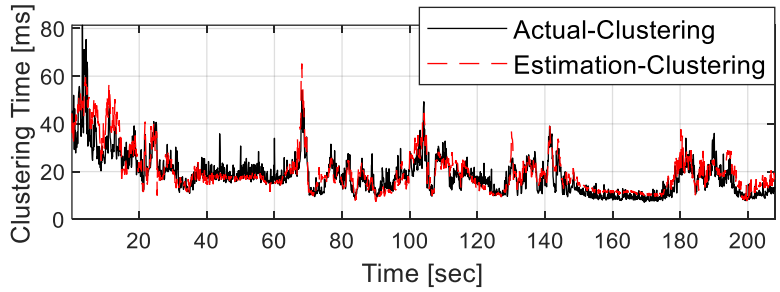
(b) Normal Probability Plot of Residuals

Figure 5.14. Residual plots of selected model M4.

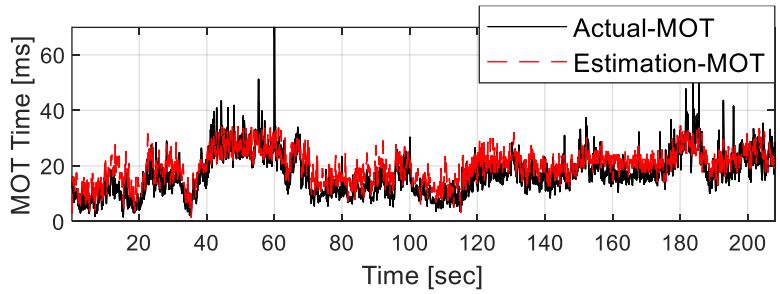
5.2.3. Validation through Data-based Simulation

In the previous two subsections, we determined a model that predicts the processing time of clustering and MOT functions that takes up the large computational load of lidar-based cognitive algorithms. In this subsection, the determined models are applied to the target perception system and evaluated by comparing the actual processing time with the estimated processing time. The results applied to the beltway at Seoul National University's Kwanak campus are described in Figure 5.15. Although there are some differences from the actual data, it can be seen that the estimation results are similar to the actual data. For detailed error analysis, residuals are shown as histograms in Figure 5.16, and the probability density function is also shown for clarity. Figure 5.16 (a) shows that the predicted time of the MOT tends to be slightly larger than the

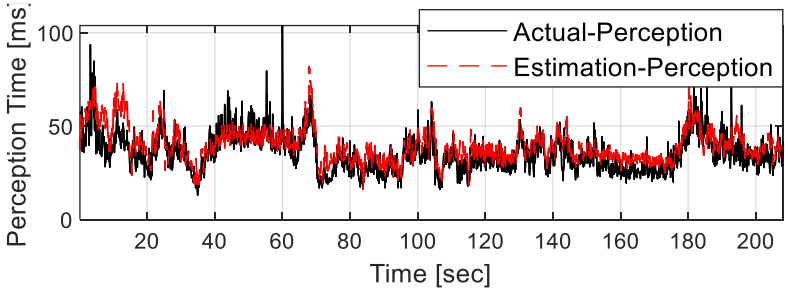
predicted time of the clustering. Table 17 shows the RMSE of the results of clustering and MOT processing time estimation. By comparing the RMSE values, it can be said that clustering has a higher accuracy of the regression model than the MOT.



(a) Clustering time estimation

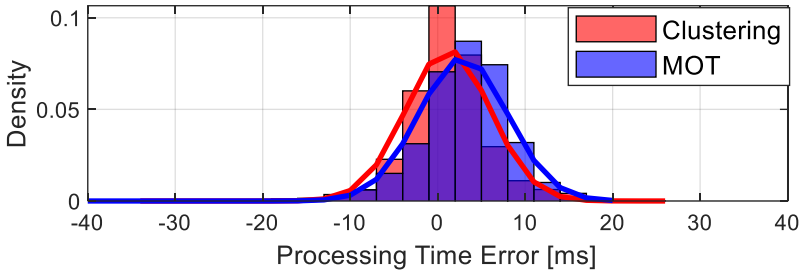


(b) MOT time estimation

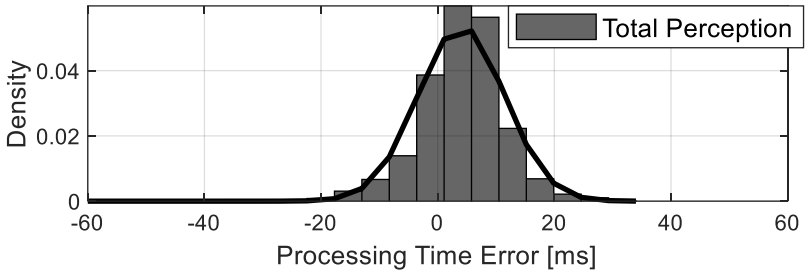


(c) Total perception time estimation

Figure 5.15. Simulation result of processing time estimation.



(a) Comparison of error distribution of clustering and MOT time estimation



(b) Error distribution of total perception time estimation

Figure 5.16. Error analysis of processing time estimation simulation.

Table 17 RMSE analysis of processing time estimation of clustering and MOT.

Function	Clustering	MOT
RMSE [ms]	4.9680	5.8616

5.3. Computational Load Management

Conventional lidar-based cognitive algorithms not only process point cloud data of limited area but also apply uniform processing to all data. When the surrounding environment is simple and the detection signal data is small, it is executed smoothly. When the surrounding environment is simple and the detection signal data is small, it is executed smoothly. Otherwise, the operation fails sometimes within the allotted calculation time, and thus the operation result of the algorithm is not normally transmitted to the next module. If the actual operation time of the module exceeds the allotted period, the operation results are not updated. Therefore, it can be said that the ability to respond immediately is reduced from the viewpoint of the entire system.

In Chapter 3, a method of performing adaptive ROI-based point cloud processing was proposed. It is more efficient and environmentally aware than existing processing techniques that process all point clouds in a batch or use map information to define areas of interest. However, there is still a possibility of exceeding the time limit due to the computational load depending on the running conditions of the system and the surrounding environment. Therefore, this section proposes a computational load management method to prevent such a failure. The concept of optimization in computer engineering means maximizing performance by improving algorithms in a limited hardware environment. There are a variety of software optimization techniques such as optimization of function call speed, optimization of operation speed,

optimization of control statements, and code optimization to reduce memory usage. Since this section focuses on the algorithm's execution cycle adherence, not on the code optimization, we develop a computational load management strategy by applying computational assessment of perception algorithm.

The computation load management proposed in this study is ultimately aimed at reducing execution time. If the predicted processing time exceeds the given execution time, it should be reduced, as mentioned earlier, since the size of the point cloud to be handled is the most significant cause due to the algorithm characteristics. The number of point clouds to be processed should be reduced by reducing the required perception ROI. The previously designed ROI can be reduced in two ways: area selection based on importance and reduction of overall design area due to deceleration control. These two methods are critical components of computational load management with processing time reduction and restriction of driving control, as shown in Figure 5.17.

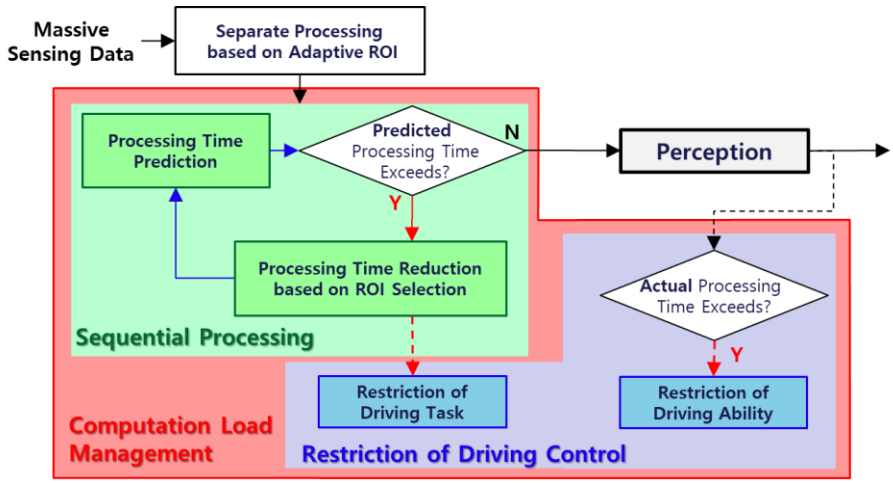


Figure 5.17. Scheme of proposed computation load management.

Sequential processing in processing time reduction is a method that excludes areas of low ROI importance step by step if the predicted computation time is not appropriate. The ROIs classified as 1st, 2nd, and 3rd are dropped from the 3rd ROI in the lowest order, and are sequentially executed until the prediction execution time satisfies the criteria. In addition, voxelization, which is 20% looser than the default, is applied before dropping the ROI.

Restriction of the driving condition is a method to increase the real-time computational reliability by limiting the behavior of the vehicle by utilizing the vehicle behavior according to the level characteristics of the adaptive ROI and the subject vehicle speed-dependent ROI characteristics. In cases where sequential processing is heavily applied, it may not be appropriate to expand the area for lane-change, thus limiting the choice that results in zone expansion for safety reasons. Besides, ROI is reduced by limiting the top speed on the road, which leads to a decrease in the number of point clouds. Since excessive limits on driving capability can cause degradation of driving performance, reasonable criteria are established and designed to be applied only if necessary.

5.3.1. Sequential Processing to Computation Load Reduction

In Section 3.1, the adaptive ROI by level is defined and applied to lidar processing. The 1st level ROI is the most essential and essential area, and its weight decreases toward the 3rd level ROI. In Section 3.2, the processing is applied differently in each area to reflect these design characteristics to maximize efficiency. Section 5.1 analyzes and identifies the resources allocated to the algorithm in a given execution environment. Nevertheless, the

environment is so complex that the allocated execution time may be exceeded if there is a lot of input data. Therefore, processing time reduction is performed by designing sequential steps using the method of estimating the algorithm execution time in advance in the previous section 5.2.

The sequential steps are constructed, as shown in Figure 5.18, using the ROI importance level concept defined in Chapter 3. The 1st level ROI is the area of the highest importance and must be recognized — the lower the level, the lower the importance. The process of computations using all the data inside the three-step ROI was proposed in Section 3.2. The processing time is reduced by reducing the size of data to be processed by sequentially excluding the lower regions except for the 1st ROI.

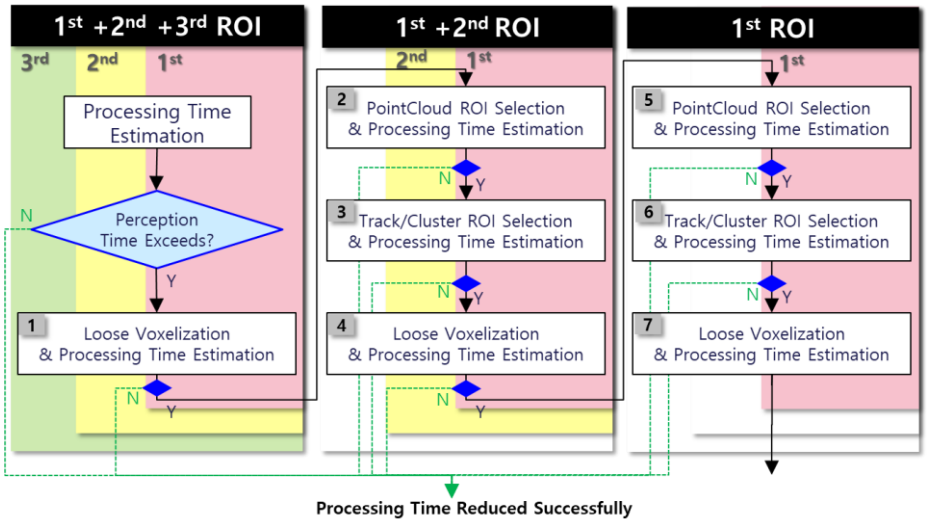


Figure 5.18. Flow chart of sequential processing time reduction.

Before applying sequential processing, it is necessary to determine the suitability of the input data using the predicted processing time. The blue diamond-shaped block illustrates this process in Figure 5.18. The criteria for appropriate predictive processing time are established and applied, taking into account the allocation time obtained in Section 5.1 and the RMSE error in each function's processing time estimation results in Section 5.2.

Sequential processing consists of three main stages: the point cloud ROI selection, the track and cluster ROI selection, and loose voxelization. In Section 5.2, the processing time reduction for the two computational processes has been performed because clustering and MOT take up most of the computational load in the lidar-based environment perception algorithm. Since the calculation time for each function can be individually reduced by changing the input data for each function, ROI selection was performed in two stages. Point cloud ROI selection directly reduces the input of the clustering function, and track and cluster ROI selection reduce the input of the MOT function. Besides, a cell configuration that is looser than the existing downsizing condition is applied to correspond to the case where the calculation time is exceeded even though the ROI selection is performed. In this study, a 20% increase in the size of the voxelization cell in subsection 3.2.2 is applied. In Figure 5.18, sequential processing is designed as a total of seven steps, and processing time prediction is performed immediately after applying each step to determine the suitability of the input data size.

5.3.2. Restriction of Driving Control

Exceeding the execution time of the modules constituting the system may be regarded as a failure from the system's point of view. According to the technical report containing the framework for test cases and scenarios of automated driving systems published by National Highway Traffic Safety Administration (NHTSA) of United States [Thorn,'18], system failure modes are classified as follows: Sensing and communication, perception, navigation and control, and Human-Machine Interface (HMI). In this case, the application consists of sensor processing, localization, and world modeling, and the item that corresponds to the objective of this study is sensor processing. The failure is summarized in the following three ways.

1) No data – Information is absent altogether.
2) Inadequate quality data – Information is of poor or degraded quality.
3) Latent data – Information is delayed or old.

If the cognition algorithm proposed in this study is not executed within the allotted period, failure of the above type may occur. The NHTSA report classifies failure mitigation into two strategies for failure: fail-operational (FO) and fail-safe (FS). FS strategies are for cases where the ADS cannot continue to operate due to a significant failure, and FO strategies are for cases where the ADS could continue to operate even in the face of failure. Since we design processing time reduction for failure release purposes from a system perspective, we have a similar orientation to the FO strategy. Among the mechanisms of FO, this study refers to degraded operations. Degraded

operation is one of the strategies to keep the system functioning in a confined environment even after a failure and limits the following items: top speed, automation level, Operational Design Domain (ODD), maneuver, and Object and Event Detection and Response (OEDR). Among these, restriction of the maneuver related to lane change and limitation of the top speed related to ROI dependent on speed, are considered in the algorithm development. Restriction of driving control is activated when certain conditions are met throughout the sequential processing and perception algorithms, as shown in Figure 5.19.

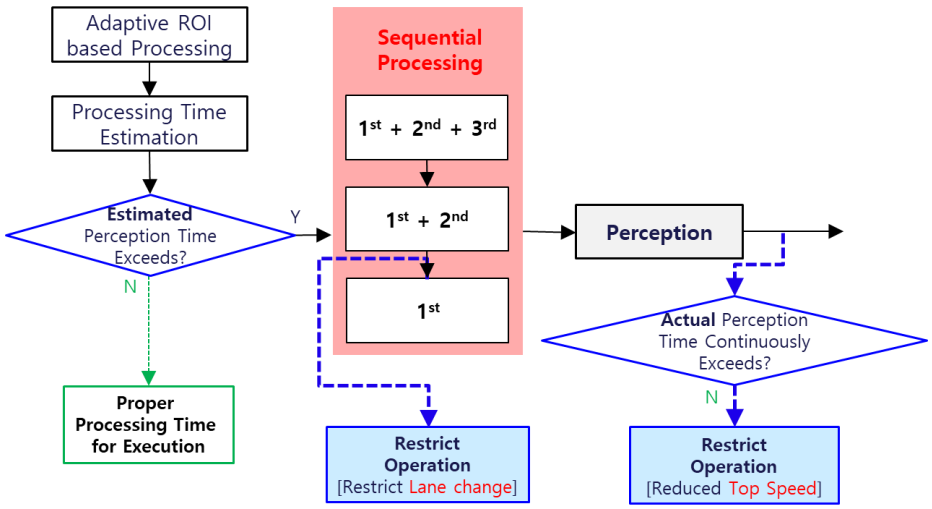
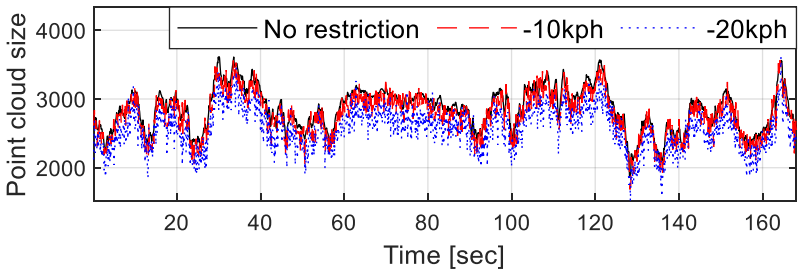


Figure 5.19. Scheme of restrict driving condition.

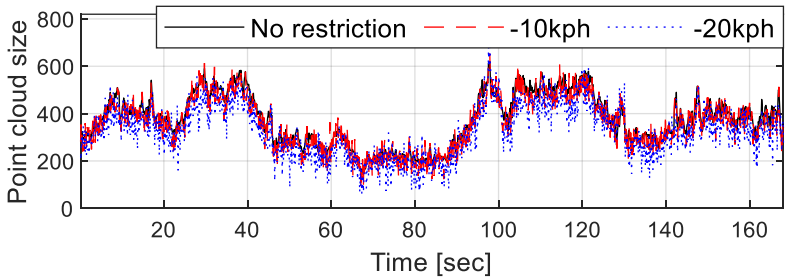
The restriction of driving control utilizes the level characteristics of adaptive ROI and the ROI characteristics dependent on the subject vehicle speed. Sequential processing excludes low-level ROI data when the predicted computation time is exceeded. At this point, the area was expanded in

subsection 3.1.2 to conduct safe lane changes. Lane change requires more ROI than normal driving conditions. If area reduction is already applied in sequential processing, it should be limited because there is no resource space to cope with area expansion. For example, if up to 2nd ROI is eliminated and only 1st ROI of data is processed, environment awareness of the three-lane area from the lane-keeping situation to both sides is performed. For safe lane changes, the environment must be extended to at least the area next to the second lane. Therefore, if only 1st ROI is selected during sequential processing, lane change is limited for reliable real-time operation of the system.

Figure 5.20 through Figure 5.22 show the results of processing at the actual driving speed for the same driving data, and the results processed according to the ROI constructed when the speed was set by 10kph and 20kph slower than the driving speed. While a decrease in speed does not always guarantee a decrease in ROI, the point cloud size that is processed tends to decrease, as shown in Figure 5.20. The execution time for each function can be seen in Figure 5.21. From that figure, the ROI change due to the reduced speed has a more significant impact on the clustering operation time than the MOT. The total computation time is depicted in Figure 5.21 (c), and a histogram in Figure 5.22 for statistical analysis. The fitting parameters are shown in Table 18 for fitting with a log-normal distribution.

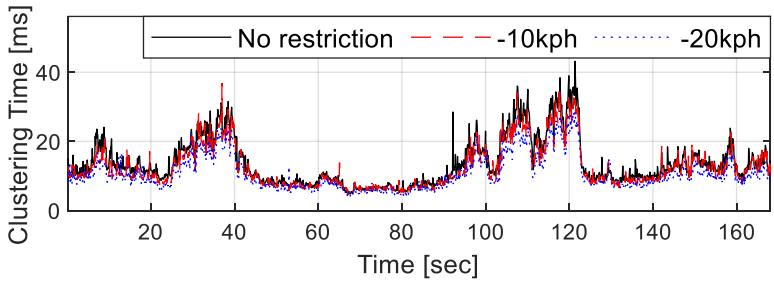


(a) Total point cloud size after ROI categorization

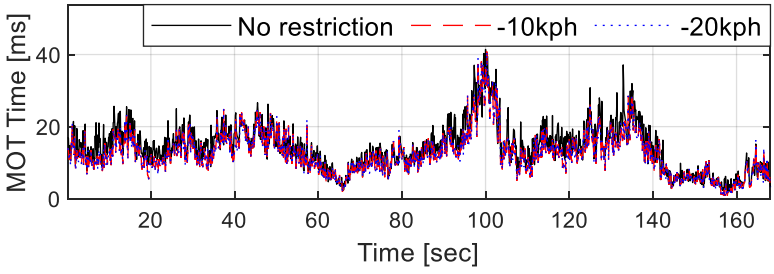


(b) Total point cloud size after separated voxelization

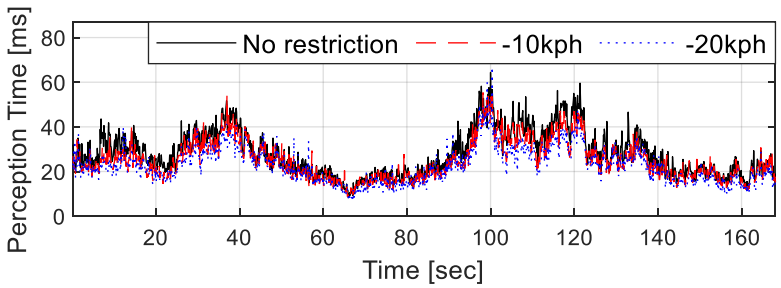
Figure 5.20. Input data comparison by applying speed restriction [-10kph, -20kph].



(a) Clustering time



(b) MOT time



(c) Total perception time

Figure 5.21. Processing time comparison by applying speed restriction.

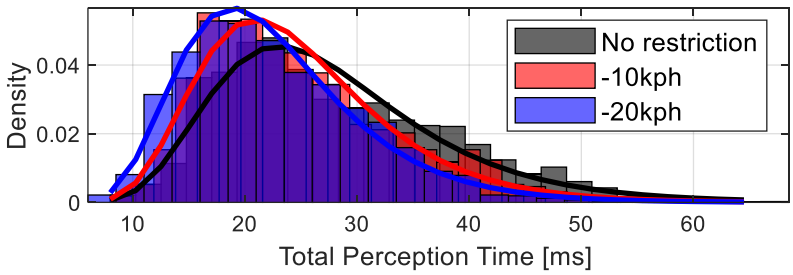


Figure 5.22. Density distribution of actual processing time by applying speed restriction [-10kph, -20kph].

Table 18 Lognormal distribution properties of perception time by top speed restriction.

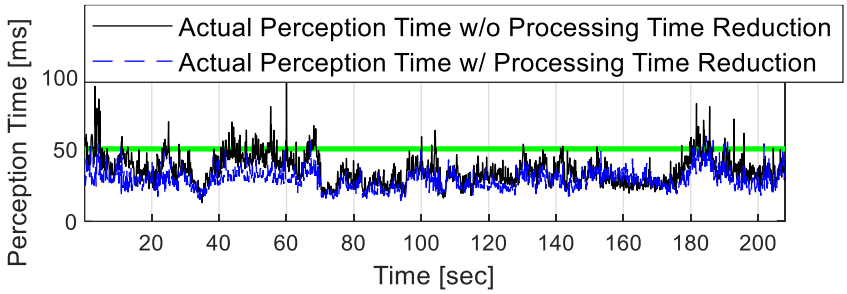
Speed Restriction Type	μ (mu)	σ (sigma)
No restriction	3.26558	0.356428
-10kph	3.15659	0.337481
-20kph	3.06129	0.350763

Despite the application of computational load management up to now, the actual execution time of the module is possible to be exceeded due to various reasons. As mentioned above, the computation load of the lidar processing is required to reduce the ROI because it is proportional to the input data size. The reduction of computation time of the perception algorithm according to the speed reduction setting was confirmed through the simulation above. The speed limiting method has a direct effect on the vehicle's behavior, and if applied indiscriminately, it is likely to be inefficient and inadequate in terms of normal driving control. In the case of self-driving systems and ADAS, there is not only a delay for each module that makes up the system, but the actuator delay is large, particularly for automobiles. If not an extreme contingency, intermittently exceeding the running time may not be serious in terms of vehicle control. Therefore, in this study, real-time performance of the perception module is ensured through a top speed limit when continuous exceedance of execution time occurs.

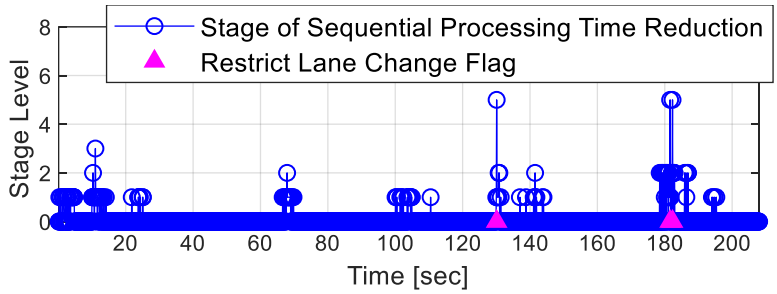
5.3.3. Validation through Data-based Simulation

The performance of the proposed sequential processing has been evaluated through driving data-based simulation. Figure 5.23 shows the simulation result of applying sequential processing to the driving data. Figure 5.23 (a) shows when sequential processing is applied and not applied to the same data. The solid green line represents the allocation maximum execution time of the perception algorithm obtained in Section 5.1, and the solid magenta line is the criterion to consider the processing time prediction error. In Figure 5.23 (a), the

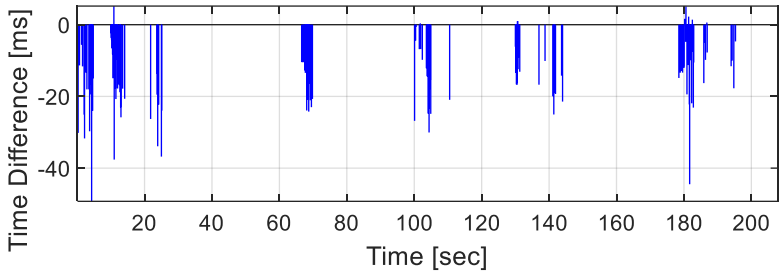
execution time without the computational load reduction indicated by the solid black line frequently exceeds the allotted time. With the computational load reduction in blue dashed line, the execution time does not exceed the allocation time most of the time. Figure 5.23 (b) shows the switch flag when processing time reduction is activated when the prediction processing time of the perception algorithm exceeds the criteria using adaptive ROI-based processing results. The reduced prediction processing time when computational load management is applied is shown in Figure 5.23 (c). The histogram is shown in Figure 5.24 for statistical analysis of the execution time of Figure 5.23 (a). Figure 5.23 (d) shows top speed restriction flag when actual processing time exceeds continuously even computational load management is applied. Figure 5.24 shows that the computational load used by existing algorithms over 40ms can be reduced to less than 40ms by applying computation load reduction, which in almost all cases results in a successful operation within the allocated cycle. These results demonstrate that the processing time reduction based computational load management proposed in this section is effective in reducing the execution time of the environment perception algorithm.



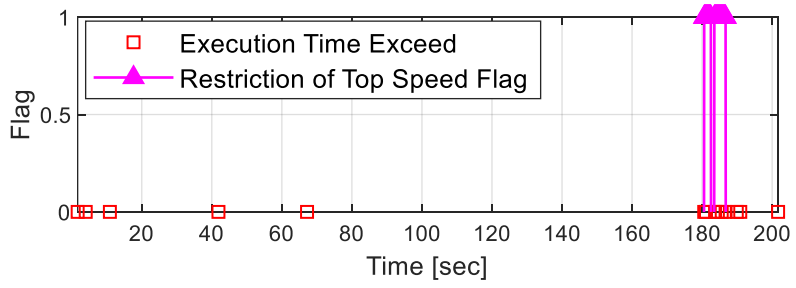
(a) Comparison of perception time between w/ and w/o processing load management



(b) Applied stage of computation load reduction and lane change restriction flag

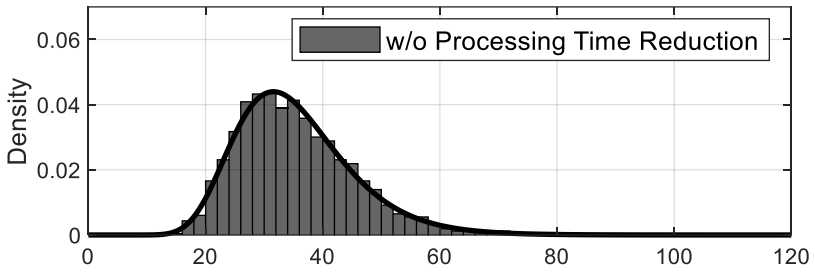


(c) Actual processing time difference between w/ and w/o processing time reduction

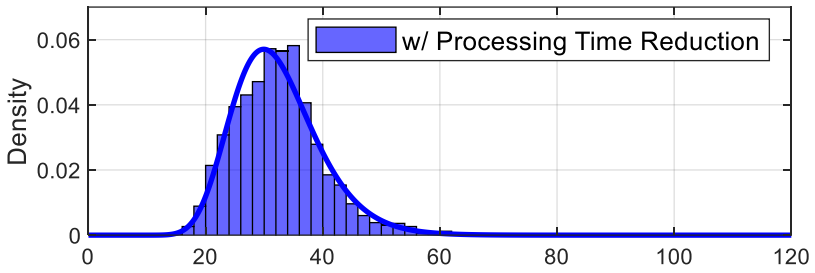


(d) Top speed restriction flag

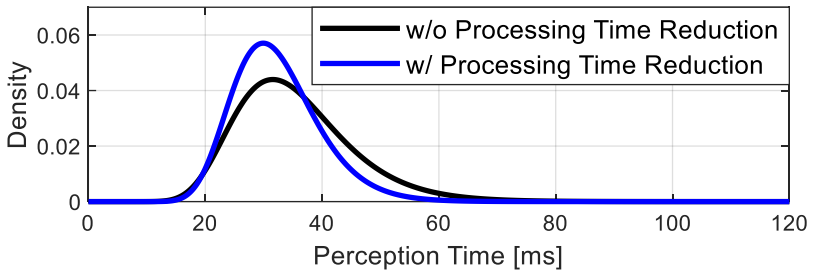
Figure 5.23. Simulation result of computational load management.



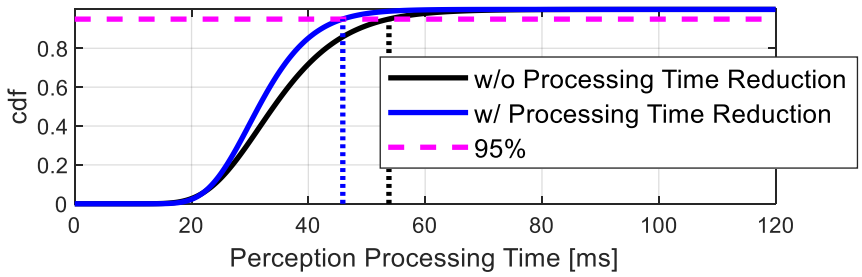
(a) Without processing time reduction



(b) With processing time reduction



(c) Comparison of fitted lognormal distribution



(d) Comparison of fitted cumulative lognormal distribution

Figure 5.24. Density distribution of actual and estimated processing time.

Chapter 6 Vehicle Tests based Performance Evaluation

The proposed algorithm is evaluated through test-data based computer simulations and actual vehicle tests. The test-data based simulation is constructed using the commercial vehicle software, MATLAB/Simulink with collected driving data. Data is collected under various driving conditions while driving on urban city roads. The automated driving system of test vehicle is configured and executed on the LabVIEW, MATLAB, and Simulink environment. The vehicle tests have been carried out on a section of the Nambu beltway of Seoul, with regular vehicles driving together. The designated test route is suitable as a test environment for evaluating algorithms because it can be considered to represent a typical urban driving environment: it includes representative facilities such as intersections, crosswalks, and median dividers and has a large amount of traffic during the day. The experimental results demonstrated that automated driving vehicle with perception algorithm based on the proposed strategy in this study successfully drives the test route.

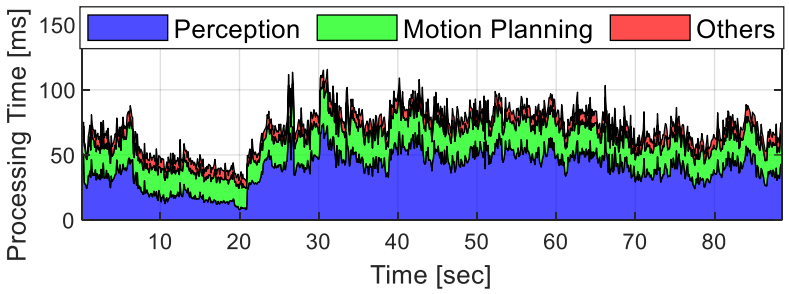
6.1. Test-data based Simulation

Analyzing the vehicle test result of the automated driving system applying the previous method, which used the batch application, it was confirmed that the operation results are not updated or delayed due to intermittent system timeout. The previous method is an approach to reduce the amount of computation by downsizing the entire data, and it is difficult to cope with the computational load due to various environmental changes. It cannot guarantee reliable operation of the system, and vehicle safety cannot be guaranteed due to the problems by applying the previous method. Vehicle safety must be guaranteed because it is directly related to human life. In this study, we proposed an algorithm that ensures system operational stability and vehicle safety at the same time by acquiring cognitive performance first by weighting the area considering the result of the automated driving plan and managing computing load. In this section, we verify that these problems can be effectively improved by applying the proposed method to two representative situations where the performance degradation of the system occurred due to the computational load problem. The performance is improved by comparing the target acceleration calculated from the clearance, TTC, and the perception result of the front driving area.

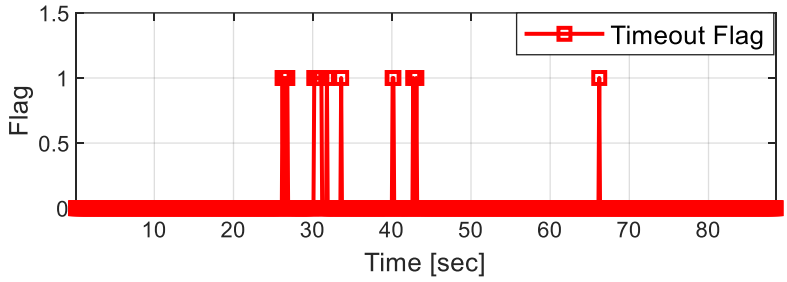
Figure 6.1 and Figure 6.3 show the results of the offline simulation by applying the method proposed in this study to the experimental data applying the previous method. Subfigure (a) shows the execution time of the entire system when the existing method is applied, and the timeout occurrences are

shown in subfigure (b). The result of applying the proposed computational load management method is shown in subfigure (c), and the variation of the computational load is relatively smaller than that of subfigure (a). Subfigure (d) shows a comparison of system runtime distributions.

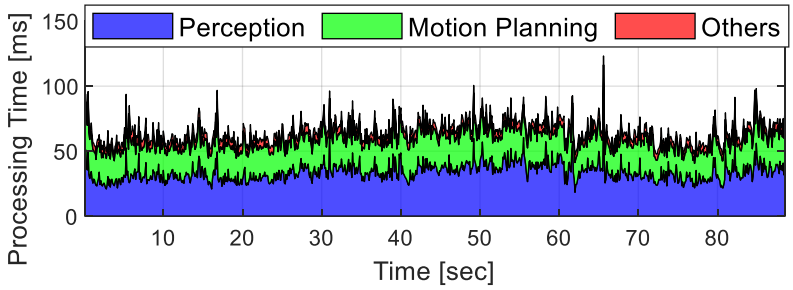
Figure 6.2 shows the case where the braking command was delayed due to a delayed update of the detection result due to a continuous eight-step timeout. Figure 6.4 also shows the situation where the update of the cognitive results fails and affects vehicle control. It indicates that braking command may be delayed by not being able to make a cut-in vehicle detection quickly, resulting in a real crash or a decline in ride quality due to a late braking command with a large degree. The proposed method can be adequately reflected in control by performing fast and accurate environment perception as indicated by the blue dashed line in Figure 6.2 by effectively managing the computational load. Besides, using the vehicle-related safety indices, Figure 6.5 shows that the proposed method achieves more reasonable risk assessment and management by accurately and reliably performing object perception than the previous method.



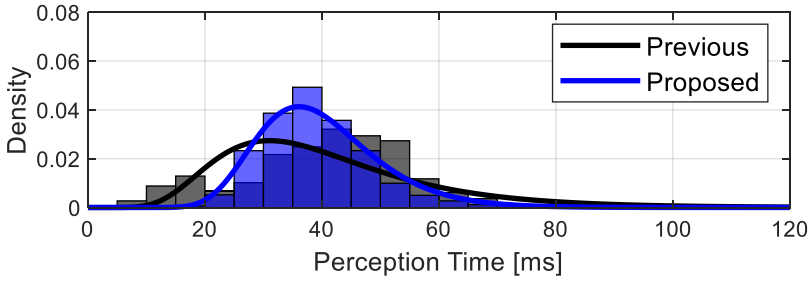
(a) Execution time of entire system for previous approach



(b) System timeout flag of previous approach



(c) Execution time of entire system for proposed approach



(d) Comparison of execution time distribution of system

Figure 6.1. The 1st simulation result by execution time analysis.

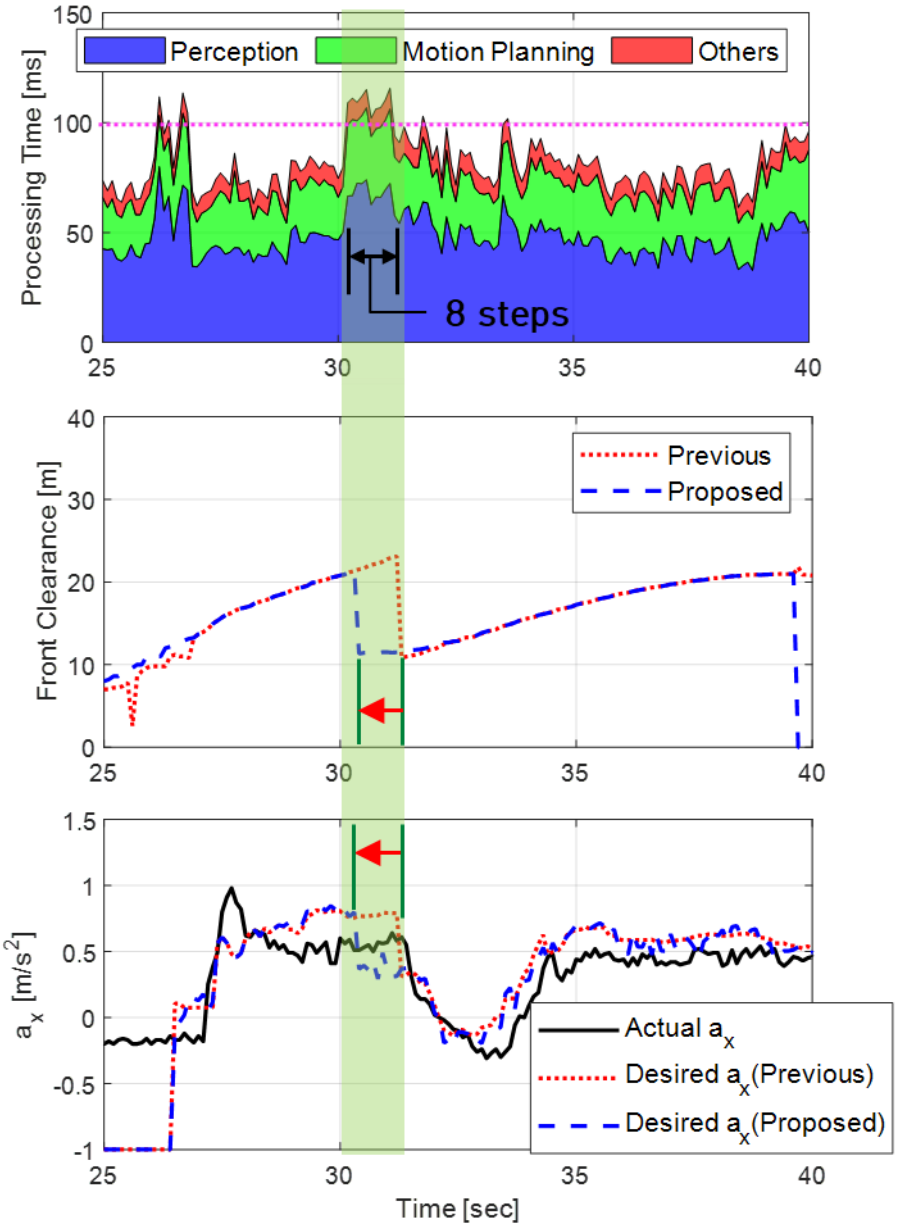
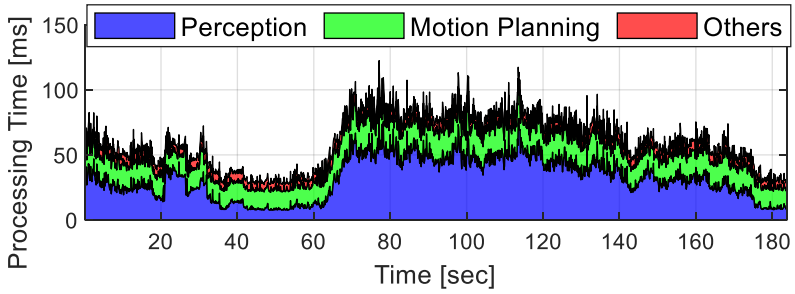
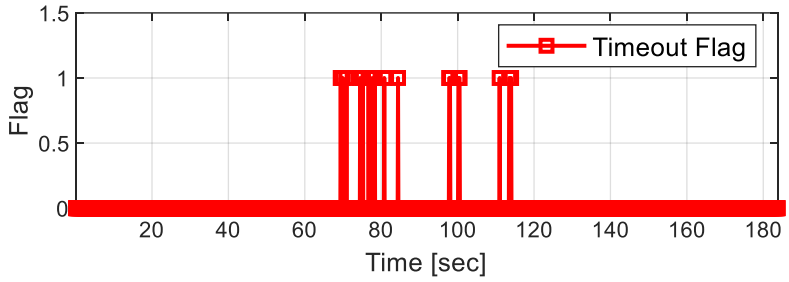


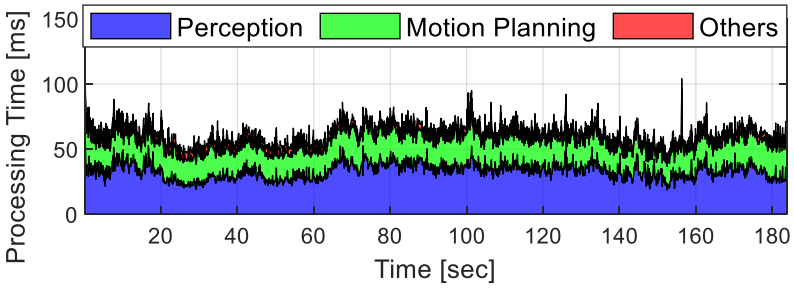
Figure 6.2. Result analysis when perception update delayed due to continuous system timeout for several cycles.



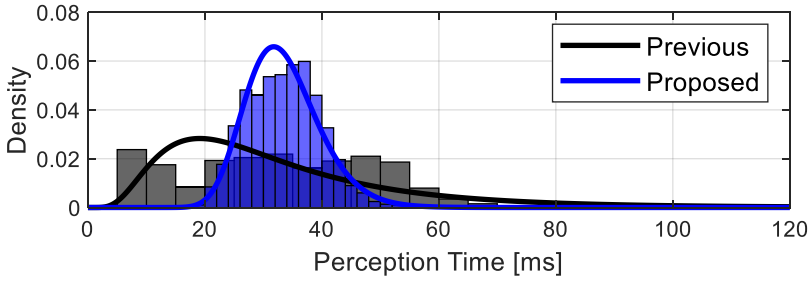
(a) Execution time of entire system for previous approach



(b) System timeout flag of previous approach



(c) Execution time of entire system for proposed approach



(d) Comparison of execution time distribution of system

Figure 6.3. The 2nd simulation result by execution time analysis.

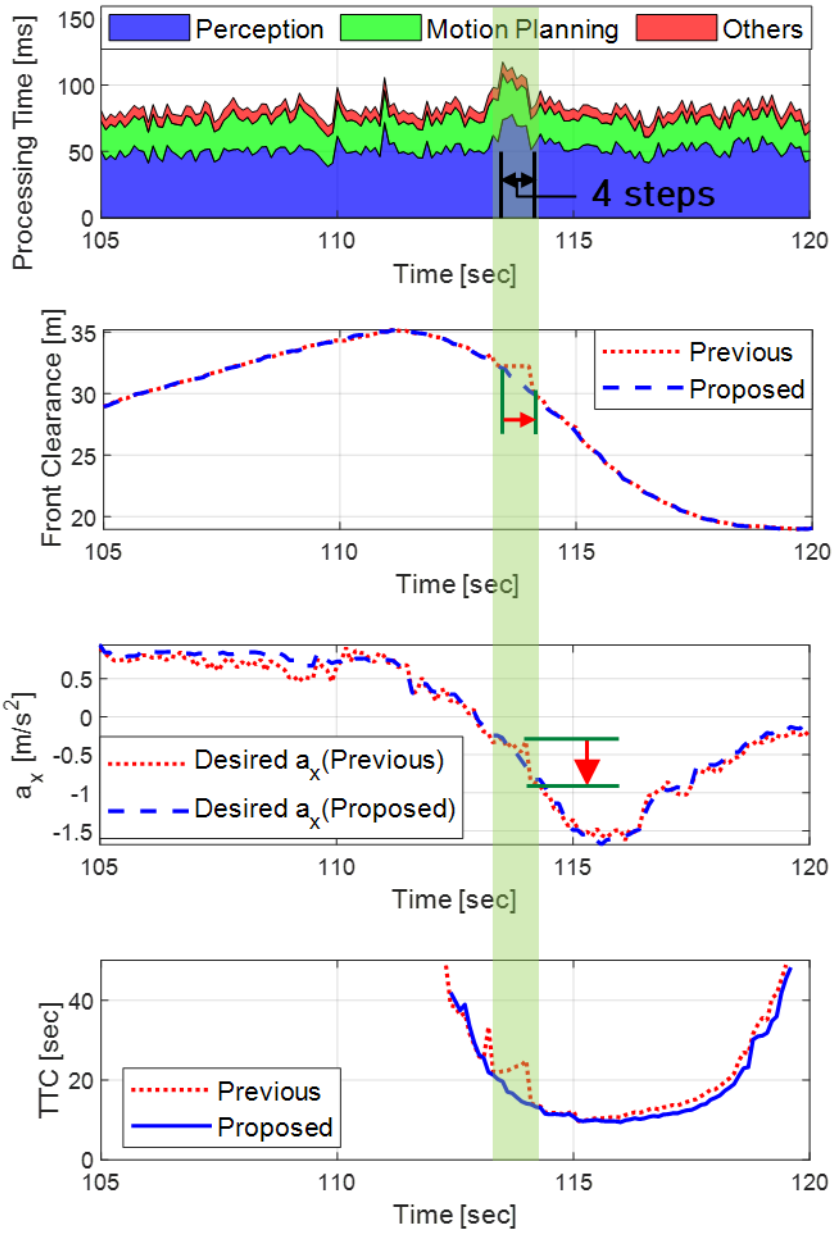
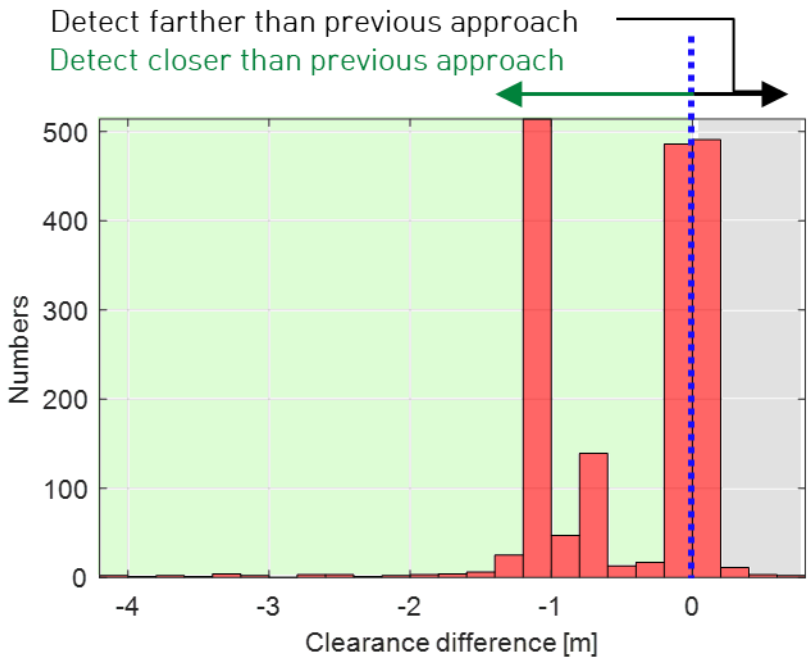
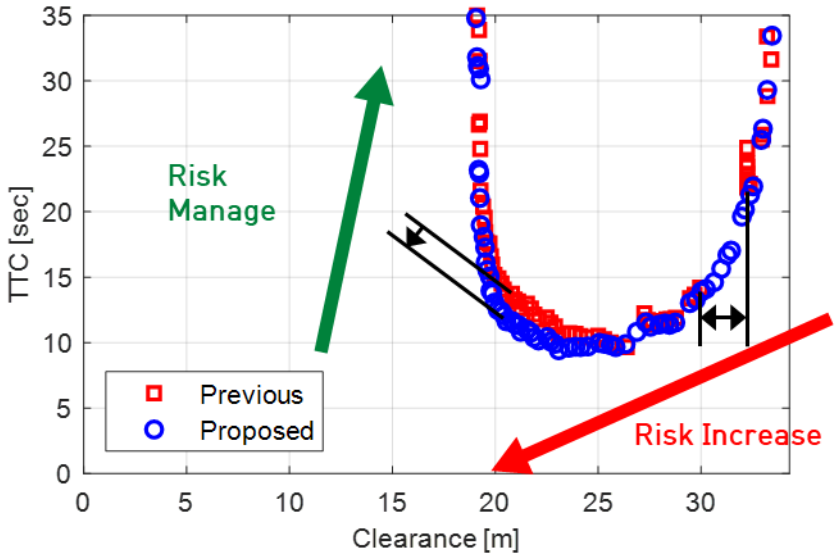


Figure 6.4. Result analysis when perception update failed due to continuous system timeout for several cycles.



(a) Histogram of front clearance changes through proposed approach



(b) Clearance-Time to collision (TTC) plane

Figure 6.5. Simulation results of vehicle safety assessment.

6.2. Vehicle Tests: Urban Automated Driving

6.2.1. Test Configuration

Vehicle experiments have been conducted at the Nambu Beltway of Seoul. The details of test roads are depicted in Figure 6.6. The designated test route is 5km long and has quite complicated environments to drive automatically. Other traffic participants should be considered because traffic on the given road is very heavy during most of the day. In addition, there are various road environments such as intersection, crosswalk, and median strip, and so forth, as shown in subfigures of Figure 6.6.

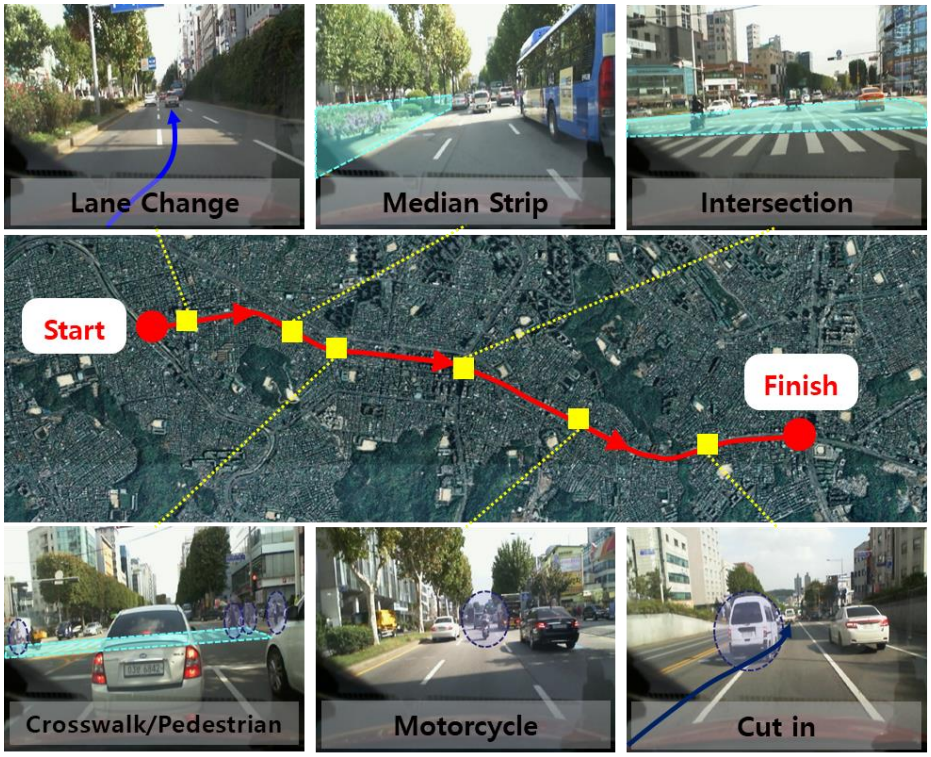


Figure 6.6. Configuration of test route in Nambu beltway (5km).

6.2.2. Motion Planning and Vehicle Control

The automated driving system that we are currently developing consists of environment perception, motion planning, and vehicle control. In this study, we dealt with research on strategies and algorithms for environment perception. To apply and operate the proposed perception strategy-based algorithm, it is implemented with an algorithm that plans and controls the vehicle behavior using perception results. This section briefly introduces the built-in motion planning and control algorithms implemented in our ADS.

The motion planning algorithm of ADS comprises three layers: dynamic environment representation, static environment representation, and motion planning optimization, as shown in Figure 6.7.

Based on the environment representation results from the perception module, the moving objects are classified, and behavior prediction is performed according to the characteristics of the classified objects. More accurate decisions and safer control can be performed through improved object behavior prediction.

The result of object behavior prediction and the constructed static obstacle map define the drivable area boundary: the free space boundary, and the drivable corridor. The free space boundary utilizes a lidar-based static obstacle map to represent the physical boundaries of the occluded area. The drivable corridor is the boundary for safe driving in traffic situations surrounding the ego vehicle. All environment information is represented on the same plane and is used to redefine the drivable corridor from the initial guess.

Based on the dynamic and static environment representation, the desired

longitudinal acceleration and desired path are determined using the Model Predictive Control (MPC) approach. Safety, dynamics, and actuator constraints are simultaneously considered to optimize the desired motion of the vehicle. We use the linear MPC based on the particle motion model. The optimal states determined by MPC are used as the desired path. The optimal acceleration input of the first prediction sampling is used as the desired acceleration.

When the desired motion of the ego vehicle is determined, the desired acceleration is applied to test the SCC module of the test vehicle to control longitudinal motion. Besides, the path-tracking controller determined the required overlay steering wheel torque to track the optimal trajectory [Jung,'14]. The required overlay steering wheel torque is applied to the MDPS system of the test vehicle.

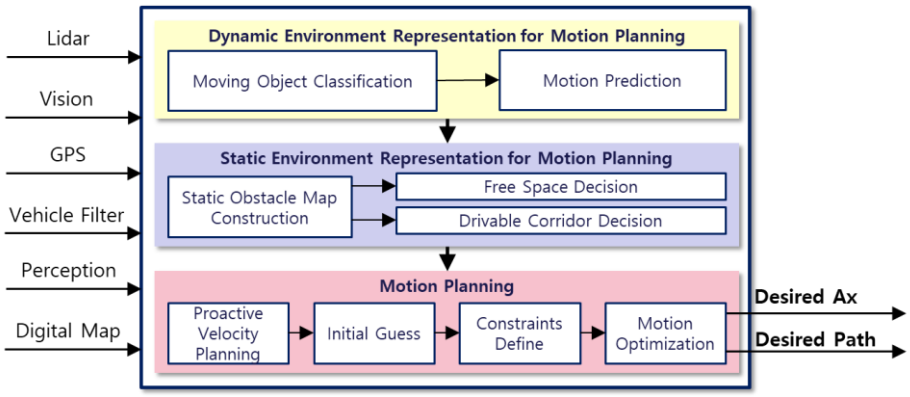


Figure 6.7. Architecture of motion planning algorithm of our automated driving system.

6.2.3. Vehicle Tests Results

Vehicle tests have been conducted several times at the Nambu beltway in Seoul, depicted in Figure 6.6. The configuration of the test vehicle was described in the previous subsection 6.2.1. The proposed algorithm mainly utilizes six multi-layer lidars, in-vehicle sensor, and front vision sensor. The proposed lidar-based environment perception algorithm operates on the LabVIEW/ MATLAB/Simulink environment of a computer installed in the test vehicle. The proposed environment perception algorithm has shown satisfactory performance, and the test results are given in Figure 6.8 through Figure 6.10.

As shown in Figure 6.8, the subject vehicle drives on urban roads with other regular traffic participants. Figure 6.8 is plotted in a body-fixed coordinate system centered on the subject vehicle marked with a blue vehicle. The areas colored in red, yellow, and green represent adaptive ROI areas for each level designed using vehicle information, road design standards, and lane information. Black dots indicate the raw data of lidar, and the point cloud data processed based on adaptive ROI is indicated by a circle of color according to the ROI area. The information representing the boundaries of the various installations and objects from the lidar sensor is extensive, as represented by scattered point clouds. The results of the proposed adaptive ROI-based processing can be confirmed to be effectively processed without distortion or loss of the boundary of the object by comparing raw data and processed points based on adaptive ROI. Objects estimated by the point cloud processing are represented by blue arrows as default and classified into In-lane objects (red

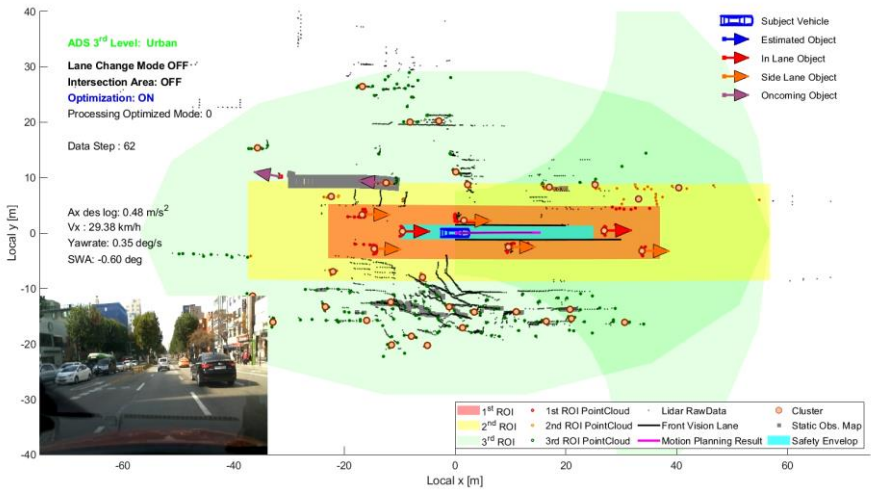
arrow), side-lane object (orange arrow), and oncoming object (violet arrow) according to the position of each object.

Figure 6.8 (a) shows the normal driving situation of the experimental vehicle along with other vehicles on a three-way road. The ROI is designed using valid lane information detected from the front vision. In the presence of a significant number of vehicles and facilities around the ego vehicle, efficient and accurate environment perception ability is required. The environment recognition result, as shown in Figure 6.8 (a), confirmed that the surrounding monitoring for safe driving is well performed. In addition, the 3rd ROI of the fan shape on the front of the vehicle can be seen acting as a backup to the forward recognition area regardless of lane information. Among the numerous point clouds acquired, the processing of data within the ROI area has shown that objects are effectively tracked. Figure 6.8 (b) describes the situation in which the vehicle in the next lane is cut-in to ego lane. Due to the high importance of processing objects near the ego lane under lane-keeping driving conditions, rapid response to the cut-in vehicle has been demonstrated. The data at 12 seconds in Figure 6.10 (a) and (b) show that the cut-in vehicle recognition results are applied to motion planning and vehicle control to achieve longitudinal control to maintain a safe distance. If there is no lane, the region of interest is designed, as shown in Figure 6.8 (c). Due to the lack of information on the surrounding road environment owing to the absence of lanes, a relatively wider area is defined than Figure 6.8 (a), and processing for this area is applied. In-lane targets marked with red arrows, as well as oncoming targets in opposite lanes marked with purple arrows, were correctly recognized.

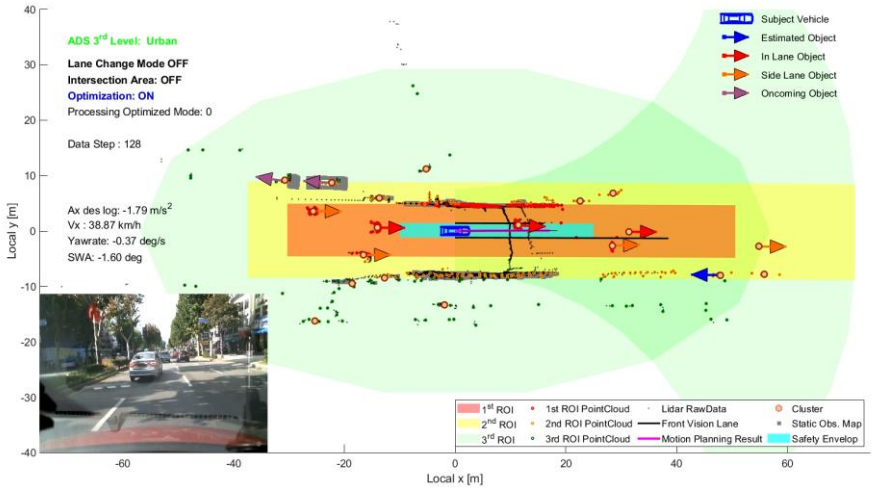
Figure 6.9 consists of subfigures that analyze the experimental results of the automated driving system applied to the test vehicle in terms of execution time evaluation. Figure 6.9 (a) shows the results of adaptive ROI based point cloud categorization and separated voxelization in Section 3.2. The processed data size for each ROI level was summed up, and as a result, downsizing was performed successfully at about 1/8 ratio. The processing stages based on the processing time estimation in Section 5.2 and computational load management in Section 5.3 are shown in Figure 6.9 (b). The higher the level, the lower the ROI, and the smaller the data to be processed. Figure 6.9 (c) indicates how much time the actual operation has decreased compared to the time previously predicted when processing time reduction is applied. Figure 6.9 (d) depicts the computation time of clustering and MOT functions that constitute the perception algorithm. It can be seen from Figure 6.9 (e) that the execution time of the environment representation algorithm to which the proposed adaptive ROI strategy is applied does not exceed the allocated time. Figure 6.9 (f) and (g) shows the execution time of the motion planning and the rest of the algorithms that comprise ADS in addition to perception. The distribution and sum of the execution time of the three major algorithm parts are shown in Figure 6.9 (h) and (i). As a result, the computation of the algorithm within the allocated computation time has been performed successfully.

Figure 6.10 shows the results of the vehicle motion planning and vehicle control in several states of the vehicle by using the environment result while driving. Using the front clearance shown in Figure 6.10 (a), motion planning calculates the target acceleration and inputs it to the vehicle's longitudinal

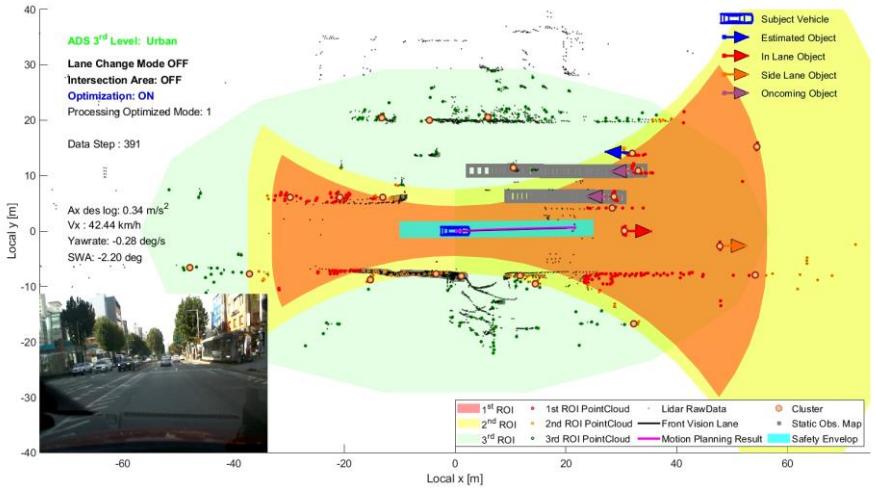
control module, which is depicted in Figure 6.10 (b). In the 12 seconds time point, the deceleration control due to the cut-in vehicle was applied and decelerated. Also, in 23 to 30 seconds, the control for the deceleration and stop was applied due to the stop of the front vehicle. Based on the desired path calculated by the motion planning algorithm, the result of applying the proper range of steering control for lane-keeping can be seen in Figure 6.10 (d). In Figure 6.11, the result of driving risk management on clearance-time to collision (TTC) plane through collision risk assessment is shown in black squares. As indicated by the red arrow, the risk of collision with the front object increases as it goes to the lower left, and appropriate driving control reduces the risk of collision as indicated by the green arrow.



(a) Normal driving with dense traffic

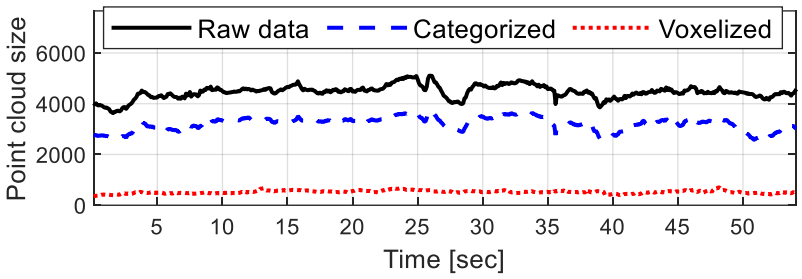


(b) Cut-in vehicle

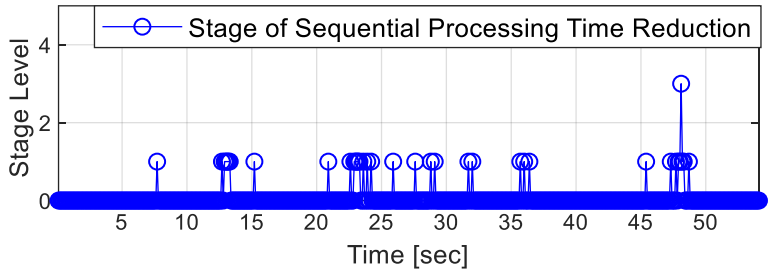


(c) No lane markings

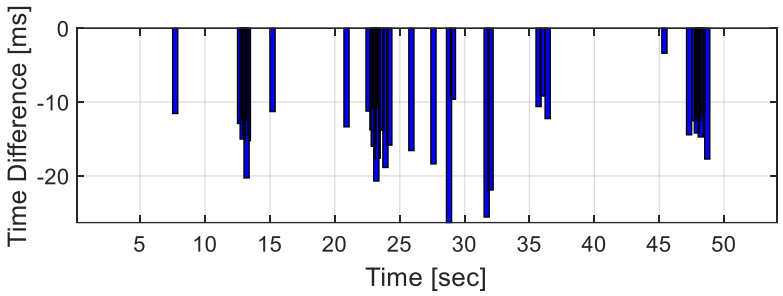
Figure 6.8. Test scenes based on adaptive ROI processing.



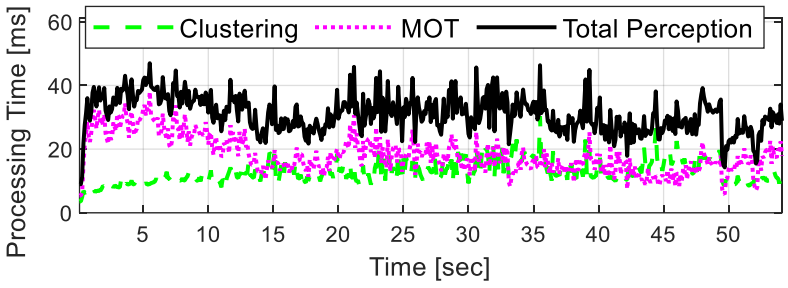
(a) Sizes of point cloud by applied process



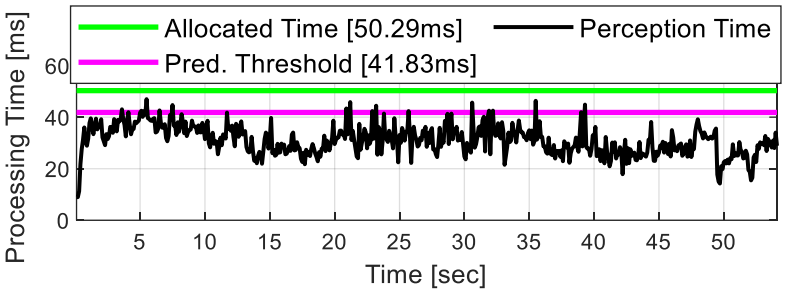
(b) Applied computation load management level



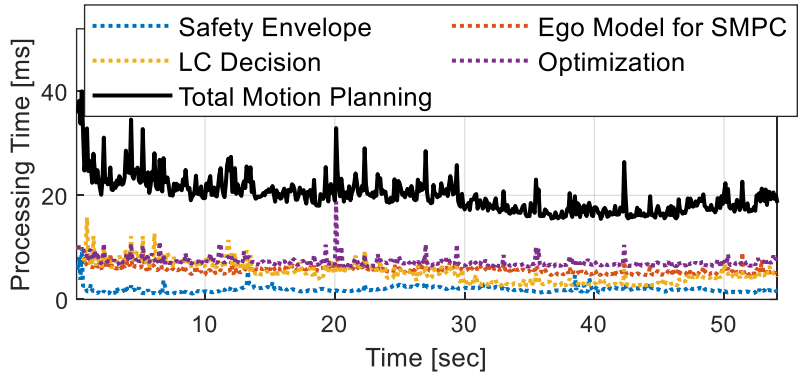
(c) Time reduction due to computation load management



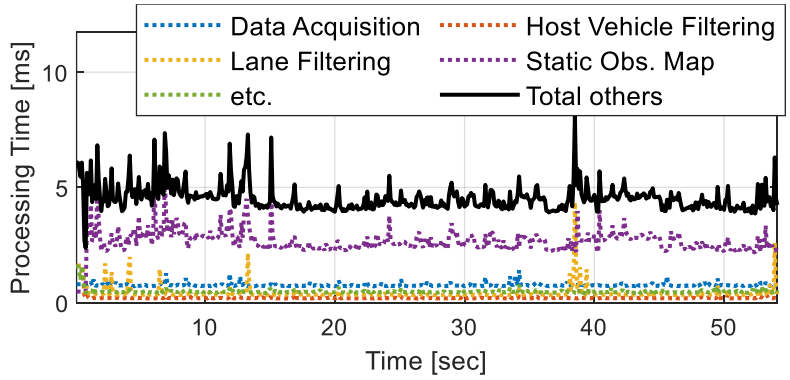
(d) Processing time for each function



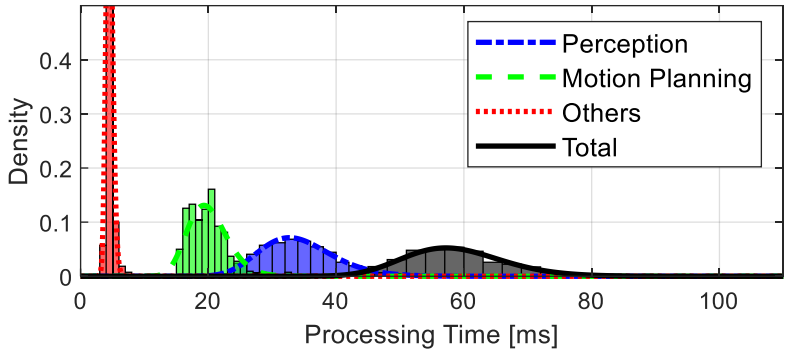
(e) Actual processing time and allocated time for perception



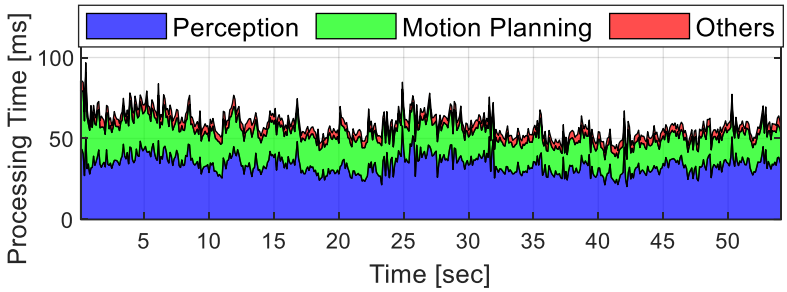
(f) Processing time of motion planning by each function



(g) Processing time of other algorithms by each function

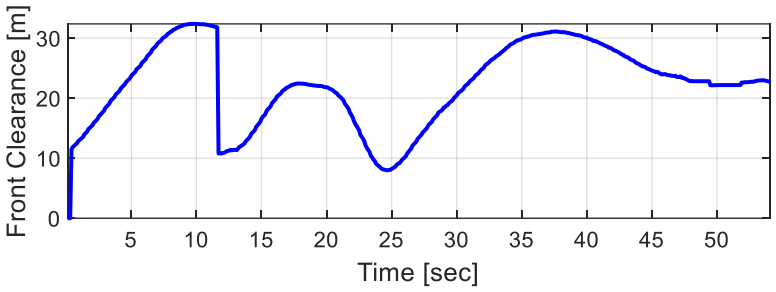


(h) Distribution of actual execution time

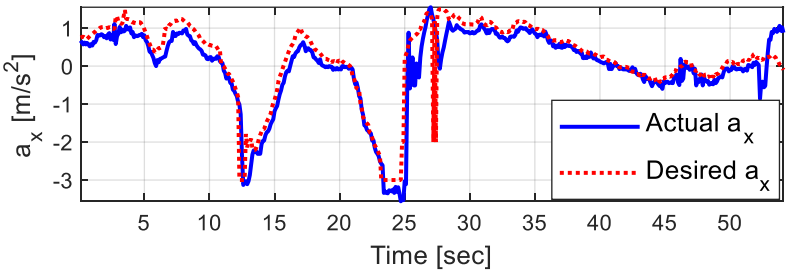


(i) Execution time summation of primary algorithms

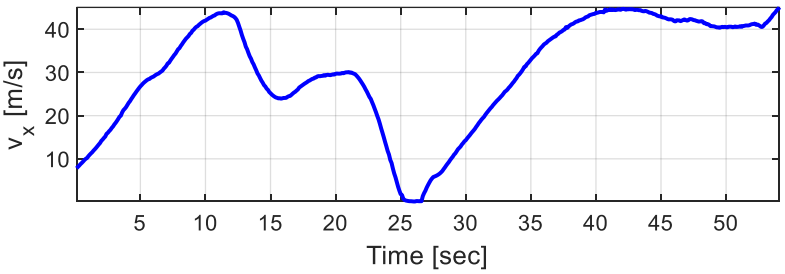
Figure 6.9. Test results of actual processing time.



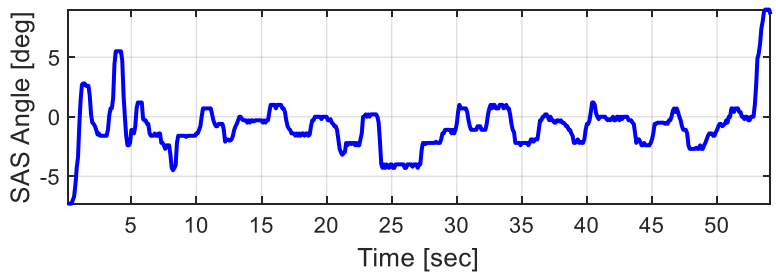
(a) Front clearance for detected in-lane object



(b) Longitudinal acceleration



(c) Actual longitudinal velocity



(d) Steering wheel angle

Figure 6.10. Test results of vehicle control.

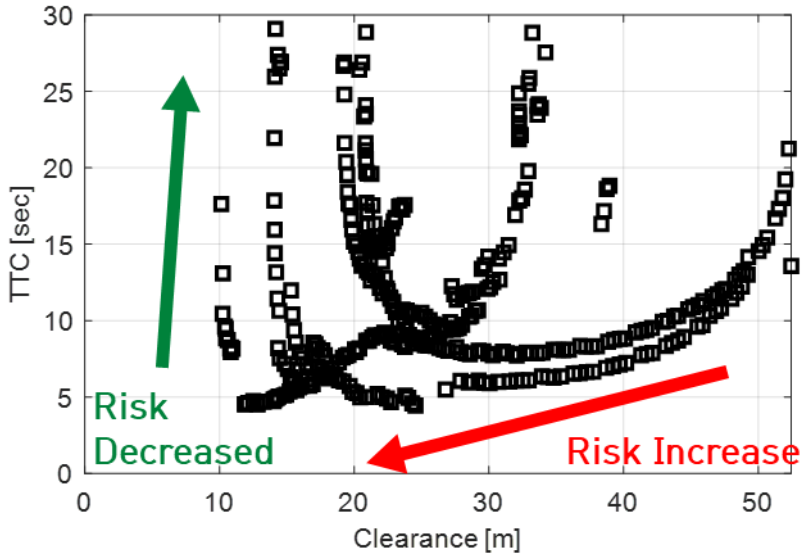


Figure 6.11. Risk management on clearance-Time to collision (TTC) plane.

Chapter 7 Conclusions and Future Works

This dissertation has proposed an efficient environment perception algorithm for a fully automated driving system that is capable of automated driving on urban roads with guaranteed safety. In this study, we focused on developing an environment perception algorithm by considering the interaction between configured modules in terms of entire system operation to secure the stable and high performance of an automated driving system. The proposed algorithm consisted of the following three steps: adaptive ROI design and processing, environment perception, and computational load management strategy. In a design of adaptive ROI, vehicle driving status and driving control solution based rational area construction has been developed. Based on defined adaptive ROI, the regional processing method, which consists of categorization, voxelization, and clustering, has been developed. With pre-processed data, environment perception algorithms for automated driving, which include time delay compensation, environment representation, and multiple object tracking, have been developed. In computational load management strategy, an analysis of allocated time for the entire automated driving system, computation load estimation based sequential processing time reduction, and driving restriction in terms of fail prevention have been developed. The developed perception algorithm computes the appropriate and sufficient environment information to

secure safe driving control.

The effectiveness of the proposed algorithm has been evaluated via test-data based simulations and actual vehicle tests. In adaptive ROI-based processing, it is found that separated processing can increase cognitive performance while reducing computational load, depending on the reasonably designed level of ROI. Moreover, motion-planning results computed for automated driving control are considered in the ROI design in order to guarantee the practical vehicle safety of the automated vehicle. The characteristics analysis of the environmental sensors is reasonably performed and experimentally verified that time delay compensation is appropriately performed to increase sensing accuracy. It is confirmed that the required environment information output from the algorithms developed for each recognition target is appropriate. In consideration of the system performance constraint determined by using human reaction time and industry standards, target hardware specification and the sensor's performance, the appropriate sampling time for automated driving system is determined to enhance safety. Resources were reasonably allocated for each configurated function through a driving data analysis that reflected the actual operating environment characteristics. To predict the computation time of complex algorithms in the target environment, multiple linear regression is performed based on the actual vehicle data, and the appropriate processing time prediction model is determined. Furthermore, it has been demonstrated that the proposed algorithm could keep the allocated execution period by applying processing time management with processing time prediction models. Based on the results, it has been shown that the proposed algorithm enhances execution

stability with respect to efficient processing.

Although the approach presented in this study has significantly improved the performance of environmental data processing, there are still elements to improve. It can be determined that the proposed approach has the potential for development. In this study, we designed ROI for rational processing under the assumption that the road surface is flat, and based on this; it performed area-specific processing and computation load management. Besides, in urban environments, various road environments, such as unformatted intersections, roundabout, and parking lots, exist, so the scope of the algorithm can be extended if these environmental characteristics are reflected in the ROI design. Although the ROI for typical driving modes has been designed and applied in this study, it is expected that a more reasonable ROI design is possible if the driving behavior characteristics for the various driving tasks, such as U-turn, joins, branches, stops, slow turns, and so forth, are reflected. In addition, if the additional environment information, such as HD map and stereo vision, obtained from other sensors is utilized, the performance may be improved by being specialized in those driving environment. Besides, in the development environment optimized for parallel operation, it is expected to maximize processing performance in terms of reduction time through not only perception and motion planning module separation but also simultaneous processing by ROI level. In addition, by analyzing various sensors and driving conditions, the optimum sampling cycle of the perception system is defined to achieve satisfactoryperformance. It is expected that the recognition and overall system performance will be improved through a variable sampling system applied to

the derived sampling period. Enhancing and verifying the proposed algorithm in the above way to achieve a high level of automated driving by extending to cover complex situations on urban roads are the topics of our future research.

Bibliography

ADARSH, S., KALEEMUDDIN, S. M., BOSE, D. & RAMACHANDRAN, K. Performance comparison of Infrared and Ultrasonic sensors for obstacles of different materials in vehicle/robot navigation applications. IOP Conference Series: Materials Science and Engineering, 2016. IOP Publishing, 012141.

ASVADI, A., PREMEBIDA, C., PEIXOTO, P. & NUNES, U. 2016. 3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes. *Robotics and Autonomous Systems*, 83**,** 299-311.

BAEK, J., HONG, S., KIM, J., KIM, E. & LEE, H. Adaptive ROI-based autonomous pedestrian detection system. 2012 12th International Conference on Control, Automation and Systems, 2012. IEEE, 635-638.

BĘDKOWSKI, J., MAJEK, K. & NUCHTER, A. 2013. General purpose computing on graphics processing units for robotic applications.

BENLIGIRAY, B., TOPAL, C. & AKINLAR, C. Video-based lane detection using a fast vanishing point estimation method. 2012 IEEE International Symposium on Multimedia, 2012. IEEE, 348-351.

BERTOZZI, M., BROGGI, A. & FASCIOLI, A. 2000. Vision-based intelligent vehicles: State of the art and perspectives. *Robotics and Autonomous systems*, 32**,** 1-16.

BISHOP, R. A survey of intelligent vehicle applications worldwide. Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE, 2000. IEEE, 25-30.

BøRCS, A., NAGY, B. & BENEDEK, C. 2017. Instant object detection in LiDAR point clouds. *IEEE Geoscience and Remote Sensing Letters*, 14**,** 992-996.

BOSSE, M. & ZLOT, R. Continuous 3D scan-matching with a spinning 2D laser. Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, 2009. IEEE, 4312-4319.

BRENNER, W. & HERRMANN, A. 2018. An overview of technology, benefits and impact of automated and autonomous driving on the automotive industry. *Digital Marketplaces Unleashed.* Springer.

CAO, V.-H., CHU, K., LE-KHAC, N.-A., KECHADI, M. T., LAEFER, D. & TRUONG-HONG, L. Toward a new approach for massive LiDAR data processing. 2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM), 2015. IEEE, 135-140.

CHO, H., SEO, Y.-W., KUMAR, B. V. & RAJKUMAR, R. R. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. Robotics and Automation (ICRA), 2014 IEEE International Conference on, 2014. IEEE, 1836-1843.

DE CECCO, M., BAGLIVO, L., ERVAS, E. & MARCUZZI, E. Asynchronous and time-delayed sensor fusion of a laser scanner navigation system and odometry. XVIII IMEKO World Congress, Metrology for a Sustainable Development, Rio de Janeiro, Brazil, 2006. 17-22.

DIETMAYER, K., KAEMPCHEN, N., FUERSTENBERG, K., KIBBEL, J., JUSTUS, W. & SCHULZ, R. 2005. Roadway detection and lane detection using multilayer laserscanner. *Advanced Microsystems for Automotive Applications 2005*. Springer.

DING, D., LEE, C. & LEE, K.-Y. An adaptive road ROI determination algorithm for lane detection. 2013 IEEE International Conference of IEEE Region 10 (TENCON 2013), 2013. IEEE, 1-4.

ELSEBERG, J., BORRMANN, D. & NþCHTER, A. Efficient processing of large 3d point clouds. 2011 XXIII International Symposium on Information, Communication and Automation Technologies, 2011. IEEE, 1-7.

FENG, D., ROSENBAUM, L. & DIETMAYER, K. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018. IEEE, 3266-3273.

FUERSTENBERG, K. C. & DIETMAYER, K. Object tracking and classification for multiple active safety and comfort applications using a multilayer laser scanner. IEEE Intelligent Vehicles Symposium, 2004, 2004. IEEE, 802-807.

GAO, H., CHENG, B., WANG, J., LI, K., ZHAO, J. & LI, D. 2018. Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics,* 14**,** 4224-4231.

GATTIS, J. & LOW, S. T. 1998. Intersection angle geometry and the driver's field of view. *Transportation research record,* 1612**,** 10-16.

GIESE, T., KLAPPSTEIN, J., DICKMANN, J. & WøHLER, C. Road course estimation using deep learning on radar data. 2017 18th International Radar Symposium (IRS), 2017. IEEE, 1-7.

GORDON, D. A. & MAST, T. M. 1970. Drivers' judgments in overtaking and passing. *Human factors,* 12**,** 341-346.

GORDON, T., SARDAR, H., BLOWER, D., LJUNG AUST, M., BAREKET, Z., BARNES, M., BLANKESPOOR, A., ISAKSSON-HELLMAN, I., IVARSSON, J. & JUHAS, B. 2010. Advanced crash avoidance technologies (ACAT) program–Final report of the Volvo-Ford-UMTRI project: safety impact methodology for lane departure warning–Method development and estimation of benefits.

GROIS, D. & HADAR, O. Complexity-aware adaptive spatial pre-processing for ROI scalable video coding with dynamic transition region. 2011 18th IEEE International Conference on Image Processing, 2011. IEEE, 741-744.

HADSELL, R., SERMANET, P., BEN, J., ERKAN, A., SCOFFIER, M., KAVUKCUOGLU, K., MULLER, U. & LECUN, Y. 2009. Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics,* 26**,** 120-144.

HALEVI, Y. & RAY, A. 1988. Integrated communication and control systems: Part I—Analysis. *Journal of Dynamic Systems, Measurement, and Control,* 110**,** 367-373.

HAN, S. H., HEO, J., SOHN, H. G. & YU, K. 2009. Parallel processing method for airborne laser scanning data using a PC cluster and a virtual grid. *Sensors,* 9**,** 2555-2573.

HARWOOD, D. W., MASON, J. M. & BRYDIA, R. E. 1999. Design policies for sight distance at stop-controlled intersections based on gap acceptance. *Transportation Research Part A: Policy and Practice,* 33**,** 199-216.

HATA, A. & WOLF, D. Road marking detection using LIDAR reflective intensity data and its application to vehicle localization. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2014a. IEEE, 584-589.

HATA, A. Y., OSORIO, F. S. & WOLF, D. F. Robust curb detection and vehicle localization in urban environments. 2014 IEEE Intelligent Vehicles Symposium Proceedings, 2014b. IEEE, 1257-1262.

HEGEMAN, G., HOOGENDOORN, S. & BROOKHUIS, K. 2004. Observations overtaking manoeuvres on bi-directional roads. *Advanced OR and AI Methods in Transportation,* 1**,** 505-510.

HESS, W., KOHLER, D., RAPP, H. & ANDOR, D. Real-time loop closure in 2D LIDAR SLAM. 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016. IEEE, 1271-1278.

HOEDEMAEKER, M. & BROOKHUIS, K. A. 1998. Behavioural adaptation to driving with an adaptive cruise control (ACC). *Transportation Research Part F: Traffic Psychology and Behaviour,* 1**,** 95-106.

HUANG, S., REN, W. & CHAN, S. C. 2000. Design and performance evaluation of mixed manual and automated control traffic. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* 30**,** 661-673.

HUTCHISON, M. C., PAUTLER, J. A. & SMITH, M. A. 2010. Traffic light signal system using radar-based target detection and tracking. Google Patents.

ISENBURG, M., LIU, Y., SHEWCHUK, J. & SNOEYINK, J. 2006. Streaming computation of Delaunay triangulations. *ACM transactions on graphics (TOG),* 25**,** 1049-1056.

ISOGAI, A., TAKAGI, K., SCHUBERT, R., RICHTER, E. & LINDNER, P. Lidar Based Lane Recognition. 16th ITS World Congress and Exhibition on Intelligent Transport Systems and Services, 2009.

IVERSON, M. A., OZGUNER, F. & POTTER, L. C. Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment. Proceedings. Eighth Heterogeneous Computing Workshop (HCW'99), 1999. IEEE, 99-111.

JO, K., JO, Y., SUHR, J. K., JUNG, H. G. & SUNWOO, M. 2015. Precise localization of an autonomous car based on probabilistic noise models of road surface marker features using multiple cameras. *IEEE Transactions on Intelligent Transportation Systems*, 16**,** 3377-3392.

JO, K., KIM, J., KIM, D., JANG, C. & SUNWOO, M. 2014. Development of autonomous car—Part I: Distributed system architecture and development process. *IEEE Transactions on Industrial Electronics*, 61**,** 7131-7140.

JUNG, C., KIM, H., SON, Y., LEE, K. & YI, K. Parameter adaptive steering control for autonomous driving. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2014. IEEE, 1462-1467.

KAEMPCHEN, N. & DIETMAYER, K. Data synchronization strategies for multi-sensor fusion. Proceedings of the IEEE Conference on Intelligent Transportation Systems, 2003. 1-9.

KAEMPCHEN, N., FRANKE, U. & OTT, R. Stereo vision based pose estimation of parking lots using 3D vehicle models. Intelligent Vehicle Symposium, 2002. IEEE, 2002. IEEE, 459-464.

KASTNER, R., MICHALKE, T., ADAMY, J., FRITSCH, J. & GOERICK, C. 2011. Task-based environment interpretation and system architecture for next generation ADAS. *Intelligent Transportation Systems Magazine, IEEE,* 3**,** 20-33.

KATO, S., TSUGAWA, S., TOKUDA, K., MATSUI, T. & FUJII, H. 2002. Vehicle control algorithms for cooperative driving with automated vehicles and

intervehicle communications. *Ieee Transactions on Intelligent Transportation Systems*, 3**,** 155-161.

KIM, B., KIM, D., KIM, K. & YI, K. High-level automated driving on complex urban roads with enhanced environment representation. 2015 15th International Conference on Control, Automation and Systems (ICCAS), 2015a. IEEE, 516-521.

KIM, B., YI, K., YOO, H.-J., CHONG, H.-J. & KO, B. 2015b. An IMM/EKF approach for enhanced multitarget state estimation for application to integrated risk management system. *IEEE Transactions on Vehicular Technology*, 64**,** 876-889.

KIM, D., KIM, B., CHUNG, T. & YI, K. 2016. Lane-level localization using an AVM camera for an automated driving vehicle in urban environments. *IEEE/ASME Transactions on Mechatronics*, 22**,** 280-290.

KIM, J. & KWEON, I.-S. Robust feature matching for loop closing and localization. 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IEEE, 3905-3910.

LEE, H., KIM, S., PARK, S., JEONG, Y., LEE, H. & YI, K. AVM/LiDAR sensor based lane marking detection method for automated driving on complex urban roads. 2017 IEEE Intelligent Vehicles Symposium (IV), 2017. IEEE, 1434-1439.

LI, P. & QIN, T. Stereo vision-based semantic 3d object and ego-motion tracking for autonomous driving. Proceedings of the European Conference on Computer Vision (ECCV), 2018. 646-661.

LIN, S.-C., ZHANG, Y., HSU, C.-H., SKACH, M., HAQUE, M. E., TANG, L. & MARS, J. The architectural implications of autonomous driving: Constraints and acceleration. ACM SIGPLAN Notices, 2018. ACM, 751-766.

LIU, M., POMERLEAU, F., COLAS, F. & SIEGWART, R. Normal estimation for pointcloud using GPU based sparse tensor voting. 2012 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2012. IEEE, 91-96.

MAGNIER, V., GRUYER, D. & GODELLE, J. Automotive LIDAR objects detection and classification algorithm using the belief theory. 2017 IEEE Intelligent Vehicles Symposium (IV), 2017. IEEE, 746-751.

MALEK-ZAVAREI, M. & JAMSHIDI, M. 1987. *Time-delay systems: analysis, optimization and applications*, Elsevier Science Inc.

MENDES, A., BENTO, L. C. & NUNES, U. Multi-target detection and tracking with a laser scanner. IEEE Intelligent Vehicles Symposium, 2004, 14-17 June 2004 2004. 796-801.

MINISTRY OF LAND, I. A. T. O. K. 2015. Rules on Structural and Facility Standards for Roads. Ministry of Land, Infrastructure and Transport of Korea.

MITSCHKE, M., WALLENTOWITZ, H. & E., S. 1991. Vermeiden querdynamisch kritischer Fahrzustände durch Fahrzustandsüberwachung (Avoidance of critical driving states in case of lateral acceleration by using driving state supervision). *VDI-Berichte*, 91.

MOON, S., MOON, I. & YI, K. 2009. Design, tuning, and evaluation of a full-range adaptive cruise control system with collision avoidance. *Control Engineering Practice*, 17**,** 442-455.

MORAS, J., CHERFAOUI, V. & BONNIFAIT, P. Credibilist occupancy grids for vehicle perception in dynamic environments. Robotics and Automation (ICRA), 2011 IEEE International Conference on, 2011. IEEE, 84-89.

MOURAGNON, E., LHUILLIER, M., DHOME, M., DEKEYSER, F. & SAYD, P. Real time localization and 3d reconstruction. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006. IEEE, 363-370.

MULLIKEN, G. H., MUSALLAM, S. & ANDERSEN, R. A. 2008. Forward estimation of movement state in posterior parietal cortex. *Proceedings of the National Academy of Sciences*, 105**,** 8170-8177.

NARANJO, J. E., GONZALEZ, C., GARCIA, R. & DE PEDRO, T. 2008. Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver. *Intelligent Transportation Systems, IEEE Transactions on*, 9**,** 438-450.

NETTO, M. S., CHAIB, S. & MAMMAR, S. Lateral adaptive control for vehicle lane keeping. American Control Conference, 2004. Proceedings of the 2004, 2004. IEEE, 2693-2698.

NEWELL, A. & CARD, S. K. 1985. The prospects for psychological science in human-computer interaction. *Human-computer interaction,* 1**,** 209-242.

NHTSA, N. H. T. S. A. 2017. 2016 fatal motor vehicle crashes: overview. *Traffic Safety Facts-Research Note (DOT HS 812 456). Online verfügbar unter https://crashstats. nhtsa. dot. gov/Api/Public/Publication/812456, zuletzt geprüft am,* 27**,** 2018.

OH, K., PARK, S., SEO, J., KIM, J.-G., PARK, J., LEE, G. & YI, K. 2019. Development of a predictive safety control algorithm using laser scanners for excavators on construction sites. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering,* 233**,** 2007-2029.

ONIGA, F. & NEDEVSCHI, S. Polynomial curb detection based on dense stereovision for driving assistance. 13th International IEEE Conference on Intelligent Transportation Systems, 2010. IEEE, 1110-1115.

PARK, S., KIM, D. & YI, K. Vehicle localization using an AVM camera for an automated urban driving. 2016 IEEE Intelligent Vehicles Symposium (IV), 2016. IEEE, 871-876.

PILA, A., SHAKED, U. & DE SOUZA, C. E. 1999. /spl Hscr//sub/spl infin//filtering for continuous-time linear systems with delay. *IEEE Transactions on Automatic Control,* 44**,** 1412-1417.

PREMEBIDA, C., MONTEIRO, G., NUNES, U. & PEIXOTO, P. A Lidar and Vision-based Approach for Pedestrian and Vehicle Detection and Tracking. 2007 IEEE Intelligent Transportation Systems Conference, Sept. 30 2007-Oct. 3 2007 2007. 1044-1049.

RAJAMANI, R. 2011. *Vehicle dynamics and control*, Springer Science & Business Media.

REN, S., HE, K., GIRSHICK, R. & SUN, J. Faster R-CNN: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 2015. 91-99.

SAHEBSARA, M., CHEN, T. & SHAH, S. L. 2007. Optimal filtering with random sensor delay, multiple packet dropout and uncertain observations. *International journal of control*, 80**,** 292-301.

SALTI, S., TOMBARI, F. & DI STEFANO, L. 2014. SHOT: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125**,** 251-264.

SAMEJIMA, S. & SEKIYAMA, K. Multi-robot visual support system by adaptive ROI selection based on gestalt perception. 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016. IEEE, 3471-3476.

SATONAKA, H., OKUDA, M., HAYASAKA, S., ENDO, T., TANAKA, Y. & YOSHIDA, T. Development of parking space detection using an ultrasonic sensor. PROCEEDINGS OF THE 13th ITS WORLD CONGRESS, LONDON, 8-12 OCTOBER 2006, 2006.

SCHREIBER, M., KNøPPEL, C. & FRANKE, U. Laneloc: Lane marking based localization using highly accurate maps. 2013 IEEE Intelligent Vehicles Symposium (IV), 2013. IEEE, 449-454.

SHALEV-SHWARTZ, S., SHAMMAH, S. & SHASHUA, A. 2016. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*.

SIMON, D. 2006. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*, John Wiley & Sons.

SINGH, M. & KAUR, A. An efficient hybrid scheme for key frame extraction and text localization in video. 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2015a. IEEE, 1250-1254.

SINGH, S. 2015b. Critical reasons for crashes investigated in the national motor vehicle crash causation survey.

SIVARAMAN, S. & TRIVEDI, M. M. 2013. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems,* 14**,** 1773-1795.

SUH, J., YI, K., JUNG, J., LEE, K., CHONG, H. & KO, B. 2016. Design and evaluation of a model predictive vehicle control algorithm for automated driving using a vehicle traffic simulator. *Control Engineering Practice,* 51**,** 92-107.

TAO, Z., BONNIFAIT, P., FREMONT, V. & IBANEZ-GUZMAN, J. Lane marking aided vehicle localization. 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 2013. IEEE, 1509-1515.

THORN, E., KIMMEL, S. C., CHAKA, M. & HAMILTON, B. A. 2018. A Framework for Automated Driving System Testable Cases and Scenarios. United States. Department of Transportation. National Highway Traffic Safety ….

THORPE, S., FIZE, D. & MARLOT, C. 1996. Speed of processing in the human visual system. *nature,* 381**,** 520.

TIDEMAN, M., VAN DER VOORT, M., VAN AREM, B. & TILLEMA, F. A review of lateral driver support systems. Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE, 2007. IEEE, 992-999.

TINGVALL, C. & HAWORTH, N. Vision Zero: an ethical approach to safety and mobility. 6th ITE International Conference Road Safety & Traffic Enforcement: Beyond, 2000.

TOLEDO, T. & ZOHAR, D. 2007. Modeling duration of lane changes. *Transportation Research Record,* 1999**,** 71-78.

TORDEUX, A. & SCHADSCHNEIDER, A. 2016. A stochastic optimal velocity model for pedestrian flow. *Parallel Processing and Applied Mathematics.* Springer.

UDACITY. 2017. *An Open Source Self-Driving Car* [Online]. Available: https://www.udacity.com/self-driving-car. [Accessed].

VANHOLME, B., GRUYER, D., LUSETTI, B., GLASER, S. & MAMMAR, S. 2013. Highly automated driving on highways based on legal safety. *Intelligent Transportation Systems, IEEE Transactions on,* 14**,** 333-347.

WANG, Z., HO, D. W. & LIU, X. 2004. Robust filtering under randomly varying sensor delay with variance constraints. *IEEE Transactions on Circuits and Systems II: Express Briefs,* 51**,** 320-326.

WANG, Z. & YANG, F. 2002. Robust filtering for uncertain linear systems with delayed states and outputs. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications,* 49**,** 125-130.

WHO, W. H. O. 2015. *Global status report on road safety 2015*, World Health Organization.

WHO, W. H. O. 2018. *Global status report on road safety 2018*, World Health Organization.

WOJKE, N. & HæSELICH, M. Moving vehicle detection and tracking in unstructured environments. Robotics and Automation (ICRA), 2012 IEEE International Conference on, 2012. IEEE, 3082-3087.

WOLCOTT, R. W. & EUSTICE, R. M. Visual localization within lidar maps for automated urban driving. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014. IEEE, 176-183.

WU, H., GUAN, X. & GONG, J. 2011. ParaStream: A parallel streaming Delaunay triangulation algorithm for LiDAR points on multicore architectures. *Computers & geosciences,* 37**,** 1355-1363.

YAMAMOTO, K., ISHIKAWA, Y. & MATSUI, T. Portable execution time analysis method. 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06), 2006. IEEE, 267-270.

YANG, B. & DONG, Z. 2013. A shape-based segmentation method for mobile laser scanning point clouds. *ISPRS journal of photogrammetry and remote sensing,* 81**,** 19-30.

YE, Y., FU, L. & LI, B. Object detection and tracking using multi-layer laser for autonomous urban driving. 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), 2016. IEEE, 259-264.

ZHANG, X., XU, W., DONG, C. & DOLAN, J. M. Efficient L-shape fitting for vehicle detection using laser scanners. 2017 IEEE Intelligent Vehicles Symposium (IV), 2017. IEEE, 54-59.

ZHANG, Y. & BAR-SHALOM, Y. 2011. Tracking move-stop-move targets with state-dependent mode transition probabilities. *Aerospace and Electronic Systems, IEEE Transactions on,* 47**,** 2037-2054.

ZHAO, F., JIANG, H. & LIU, Z. Recent development of automotive LiDAR technology, industry and trends. Eleventh International Conference on Digital Image Processing (ICDIP 2019), 2019. International Society for Optics and Photonics, 111794A.

ZHU, C. & RAJAMANI, R. 2006. Global positioning system-based vehicle control for automated parking. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering,* 220**,** 37-52.

ZIEGLER, J., BENDER, P., SCHREIBER, M., LATEGAHN, H., STRAUSS, T., STILLER, C., DANG, T., FRANKE, U., APPENRODT, N. & KELLER, C. G. 2014. Making Bertha drive—An autonomous journey on a historic route. *IEEE Intelligent transportation systems magazine,* 6**,** 8-20.

초 록

자율 주행 시스템의 차량 안전을 위한 적응형 관심 영역 기반 효율적 환경 인지

전 세계적으로 자동차 사고로 120 만 명이 사망하기 때문에 교통 사고에 대한 기본적인 예방 조치에 대한 논의가 진행되고 있다. 통계 자료에 따르면 교통 사고의 94 %가 인적 오류에 기인한다. 도로 안전 확보의 관점에서 자율 주행 기술은 이러한 심각한 문제를 해결하는 방법으로써 관심이 높아졌으며, 연구 개발을 통해 단계적 상용화가 이루어지고 있다. 주요 자동차 제조업체는 이미 차선 유지 보조장치 (LKAS: Lane Keeping Assistant System), 적응형 순항 제어 시스템(ACC: Adaptive Cruise Control), 주차 보조 시스템 (PAS: Parking Assistance System), 자동 긴급 제동장치 (AEB: Automated Emergency Braking) 등의 첨단 운전자 보조 시스템 (ADAS)을 개발하고 상용화하였다. 또한 Audi의 Audi AI Traffic Jam Pilot, Tesla의 Autopilot, Mercedes-Benz의 Distronic Plus, 현대자동차의 Highway Driving Assist 및 BMW의 Driving Assistant Plus 와 같은 부분 자율 주행 시스템이 출시되고 있다. 이러한 부분 자율 주행 시스템은 여전히 운전자의 주의가 수반되어야 함에도 불구하고, 안전성을

크게 향상시키는 데 효과적이기 때문에 지속적으로 그 수요가 증가하고 있다.

최근 몇 년간 많은 수의 자율주행 사고가 발생하였으며, 그 빈도수가 빠르게 증가하여 사회적으로 주목받고 있다. 차량 사고는 인명 사고와 직접적으로 연관되기 때문에 자율 주행 차량의 사고들은 자율 주행 기술 신뢰성의 저하를 야기하여 사회적인 불안감을 키운다. 최근 자율 주행 관련 사고들로 인해, 자율주행 차량의 안전성의 보장이 더욱 강조되고 있다. 따라서 본 연구에서는 자율 주행 차량의 거동 제어를 고려하여 자율 주행 시스템 관점에서 차량의 안전성을 우선적으로 확보하는 접근 방식을 제안한다.

또한 자율주행 기술 개발은 단순하게 운전을 대체하는 기술이 아니라, 첨단기술의 집약 체로써 산업적으로 매우 큰 파급력을 가진다고 전망된다. 현재 자율주행 시스템은 기존 자동차 산업의 고전적인 틀에서 확장되어, 다양한 분야의 관점에서 주도적으로 개발이 진행되고 있다. 자율 주행은 다양한 기술의 복합적인 결합으로 구성되기 때문에, 현재 각기 다른 다양한 환경에서 개발이 진행 중이며, 아직 표준화되어 있지 않은 실정이다. 대부분 각 모듈 단위의 지엽적인 성능향상을 추구하는 경향이 있으며, 구성 모듈 간 관계가 고려된 전체 시스템 단위의 접근방식은 미흡한 실정이다. 세부 모듈 단위의 지엽적인 연구 개발은 통합 시, 모듈 간 상호작용으로 인한 영향으로 시스템 관점에서 적절한 성능을 확보하기 어려울 수 있다. 각 모듈의 성능만을 고려한 일방적인 방향의 연구는 한계가 명확하며, 연관된 모듈들의 특성을 고려하여

반영할 필요가 있다. 따라서 자율주행 전체 시스템의 관점에서, 차량 안전을 우선적으로 확보하고 전체 성능을 극대화하는 효과적인 접근 방식을 본 연구에서 제안하고자 한다.

본 연구는 자율 주행 시스템의 안정적이고 높은 성능을 확보하기 위해 전체 시스템 작동 측면에서 구성된 모듈 간의 상호 작용을 고려하여 효율적인 환경 인식 알고리즘을 개발하는데 중점을 둔다. 실질적인 관점에서 효과적인 정보 처리를 수행하고 차량 안전을 확보하기 위해 적응형 관심 영역 (ROI) 기반 계산 부하 관리 전략을 제안한다. 차량의 거동 특성, 도로 설계 표준, 추월 및 차선 변경과 같은 주변 차량의 주행 특성이 적응형 ROI 설계 및 주행 상황에 따른 영역 확장에 반영된다. 또한, 자율 주행 차량의 실질적인 안전을 보장하기 위해 ROI 설계에서 자율 주행 제어를 위한 거동 계획 결과가 고려된다. 보다 넓은 주변 영역에 대한 환경 정보를 확보하기 위해 라이다 데이터는 설계된 ROI별로 분류되며, 영역별 중요도에 따라 연산 과정이 분리되어 수행된다. 목표 시스템을 구성하는 모듈 별 연산 시간이 측정된 데이터 기반으로 통계적으로 분석된다. 운전자의 반응 시간, 산업 표준, 대상 하드웨어 사양 및 센서 성능을 기반으로 결정된 시스템 성능 조건을 고려하여, 안전성을 확보하기 위한 자율 주행 시스템의 적절한 샘플링 주기가 정의된다. 데이터 기반 다중 선형 회귀 분석은 인식 모듈을 구성하는 함수 별 실행 시간을 예측하기 위해 적용되며, 안정적인 실시간 성능을 보장하기 위해 적응형 ROI를 기반으로 자율 주행 안전에 필요한 데이터를 선택적으로 분류하여 연산 부하가 감축된다. 연산 부하 평가 관리에서 환경 인지 모듈과 전체

시스템의 연산 부하가 대상 환경에서의 적절성을 평가하고, 연산 부하 관리에 문제가 있을 때 자율 주행 차량의 거동을 제한하여 시스템 안정성을 유지함으로써 차량 안전성을 확보한다.

제안된 자율주행 인지 전략 및 알고리즘의 성능은 데이터 기반 시뮬레이션 및 실차 실험을 통해 검증되었다. 실험 결과를 통해 제안된 환경 인식 알고리즘은 자율 주행 시스템을 구성하는 모듈 간의 상호 작용을 고려하여 도심 도로 환경에서 자율 주행 차량의 안전성과 시스템의 안정적인 성능을 보장할 수 있음을 확인하였다.

주요어: 자율 주행 시스템, 적응형 관심 영역, 라이다 프로세싱, 환경 인지, 차량 안전, 연산 부하 관리

학 번: 2014-22484