가우시안 프로세스 회귀모델을 이용한 실시간 차량 거동 학습 기반 자율주행차량 조향제어

Online Vehicle Motion Learning Based Steering Control for an Automated Driving System Using Incremental Sparse Spectrum Gaussian Process Regression

지도교수 이 경 수

이 논문을 공학박사 학위논문으로 제출함

2019 년 10 월

서울대학교 대학원

기계항공공학부

고 영 일

고영일의 공학박사 학위논문을 인준함

2019 년 12 월

위 원 장 : 차 석 원 (서명)

부위원장 : 이 경 수 (서명)

위 원 : 이 창 건 (서명)

위 원 : 강 연 식 (서명)

위 원 : 송 봉 섭 (서명)

공학박사학위논문

가우시안 프로세스 회귀모델을 이용한 실시간 차량 거동 학습 기반 자율주행차량 조향제어

Online Vehicle Motion Learning based Steering Control for an Automated Driving System using Incremental Sparse Spectrum Gaussian Process Regression

2020년 2월

서울대학교 대학원

기계항공공학부

고 영 일

Abstract

Online Vehicle Motion Learning based Steering Control for an Automated Driving System using Incremental Sparse Spectrum Gaussian Process Regression

Youngil Koh

School of Mechanical and Aerospace Engineering

The Graduate School

Seoul National University

An automated driving vehicle has come into the spotlight with the expectation to be an excellent solution to various traffic-related problems, which our society is faced with today. From recent advances in the environment perception technologies and the real-time computing capability, the automated driving vehicle is approaching to our real-life rapidly. Numerous research teams around the world are expanding the driving situations that an automated driving vehicle can handle. Consequentially, the automated driving technology is continuously advancing to conquer all driving scenarios from the highway to complex-urban.

To deal with chaotic scenarios in complex-urban, a steering control part in the automated driving system has to secure reliable performance within the full range of steering wheel angle because the automated driving vehicle has to be capable of dealing with the U-turn or mini-roundabout scenarios in inner-city-street.

For that reason, this dissertation focused on developing the steering control algorithm, which is capable of tracking the desired trajectory from the near-straight to the minimum turning radius.

The proposed steering control algorithm compensates for the current error and the predicted future error both. In order to determine the predicted future error, the proposed algorithm predicts the future driving trajectory of a vehicle from the current states.

The vehicle model for the prediction of the future driving trajectory is identified incrementally using the driving data stream by the incremental sparse spectrum Gaussian process regression (ISSGPR) method. The proposed online vehicle model learning approach contributes to the practicality because the prior knowledge of the vehicle dynamics is unnecessary.

The proposed steering controller has been validated via vehicle tests and simulations with various challenging scenarios. It was found that the compensation of the predicted future error is effective to track the desired trajectory from the straight to the minimum turning radius. The proposed steering controller shows better tracking performance in comparison with the pure-pursuit and model predictive control approach both.

It is also validated that the proposed online vehicle model learning approach is a quite practical way to predict the motion of a vehicle from the driving data stream. The conventional way of the modeling parameter identification process, which is time-consuming, is unnecessary with the proposed approach. Simulations with open-loop steering input scenario from ISO/TR 8725:1988 has been conducted to quantify the online learning performance. Various vehicle tests show that the behavior of the online learned vehicle model is quite similar to that of the actual vehicle model.

Keywords: Automated driving vehicle, Steering Control, Trajectory Tracking, Incremental sparse spectrum Gaussian process, Online vehicle motion learning, Synthetic input technique, Model reference adaptive control, U-turn scenario

Student Number: 2013-20640

Contents

List of Figures

Chapter 1

Introduction

1.1. Research Motivation and Background

It has been quite a long time since the concept and usefulness of an automated driving vehicle began to be introduced to the public. In the present, some video clips which show seamless autonomous driving can be found easily on the internet. Accordingly, the first question from the public is when they can travel safely to anywhere with the automated driving car in real.

NAVIGANT Research has issued the report, which contains the ranking of 20 leading companies who are developing an automated driving system in the first quarter of every year. The companies are evaluated from several criteria: strategy and the execution ability of the companies. Although it is uncertain that when people can utilize the automated driving car in daily life, the report helps people guess how far away we are from a driverless future in our daily life.

The report divides the 20 companies into the four groups: LEADERS, CONTENDERS, CHALLENGERS, FOLLOWERS in sequence from top to bottom. In 2019, only three companies belonged to the LEADERS group. Waymo, GM Cruise, and Ford won the first, the second, and the third place, respectively, in the 1st quarter of 2019. Figure 1.1 and Figure 1.2 denote the leaderboard grid from NAVIGANT research in 2018 and 2019, respectively. Compared to 2018, the number of companies that belong to the LEADERS group is decreased in 2019. Only three companies were left in the LEADERS group, and several companies were crowded out from the LEADERS group.

The report implies that staying in the LEADERS group needs continuous advancements in preparing the distribution of the automated driving system. Waymo and GM Cruise that are considered to be top-notch have been operating and testing their automated driving vehicles in an urban city for 24 hours. They are considered to solve the problem that exists in the real traffic environment, and they have been releasing the video clips, which show the driving performance of their automated driving vehicle in an urban area, to the public steadily.

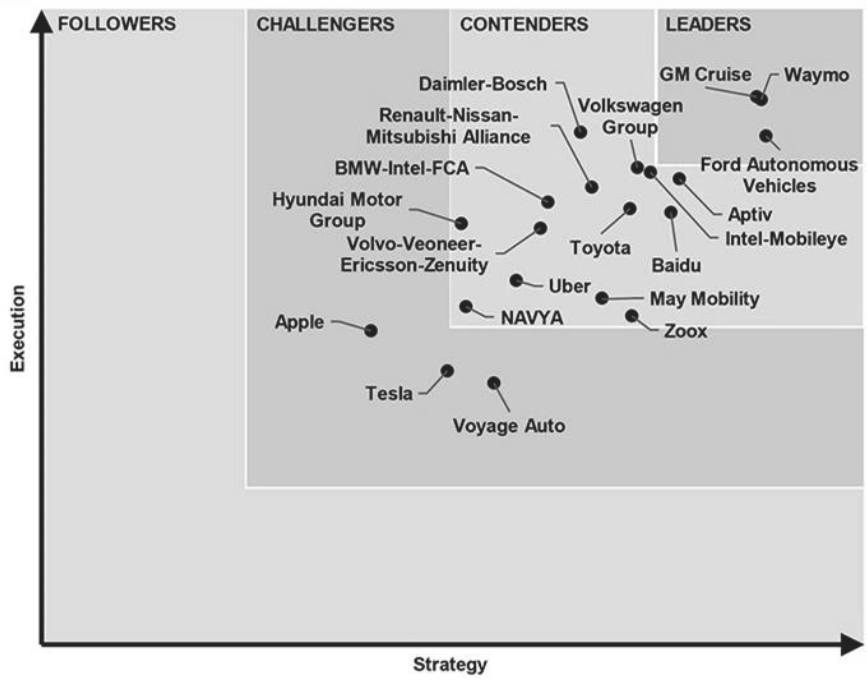

Figure 1.1. Assessment of Strategy and Execution for 20 Companies Developing Automated Driving Systems in 2019 (Navigant-research, 2019)

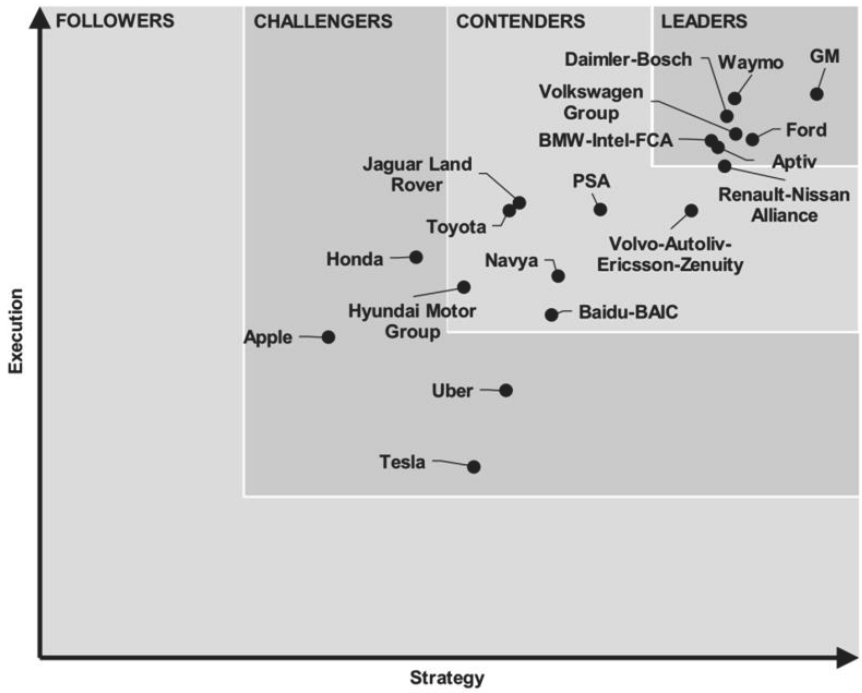

Figure 1.2. Assessment of Strategy and Execution for 19 Companies Developing Automated Driving Systems in 2018 (Navigant-research, 2019)

When evaluating the level of the automated driving system, the SAE J3016 standard is frequently referred. The SAE J3016 standard defines the six levels of driving automation (Committee, 2018). According to the SAE J3016 standard, the system with SAE level 3 and above is considered as the automated system. The system with below SAE level 3 is classified into the driver-assist system. Therefore, when the above SAE level 3 system is activated, the human in the driver's seat is not considered to be driving.

Most automakers seem to prefer to release the automated driving system that is gradually enhanced from a driver-assistance-system. The automation level of the commercialized automated driving systems, which are optionally equipped in the mass-produced vehicles, is considered as somewhere between 2 or 3. The automated driving system released by the automakers commonly operates in a structured area such as the highway or motorway.

Most IT companies and the autonomous driving technology start-ups are trying to develop the SAE level 5 fully-automated driving system at once without the gradual advancement in the level of driving automation.

In view of commercialization, the driving scenario that the automated driving system can cover in the present is somewhere between the motorway and the complex urban. It is more challenging to operate an automated driving vehicle in the chaotic urban than in the motorway. Accordingly, the automated driving system has to be able to cover more challenging scenarios for autonomous driving in a complex urban city. Notably, the steering control part in the automated driving system is required to be operated in the lock-to-lock range.

Figure 1.3 presents the driving trajectory and the steering wheel angle data that measured from manual driving. A route is from Seoul Nat'l University to Pyeongchon station in Anyang-si. The driving route consists of a motorway and a simple urban course. The steering wheel angle data is represented in a histogram. The histogram shows the usage range of the steering wheel angle for the motorway and simple urban driving. About 95% of the used steering wheel angle ranges within 40 degrees and the steering wheel angle hardly exceeds 90degrees.

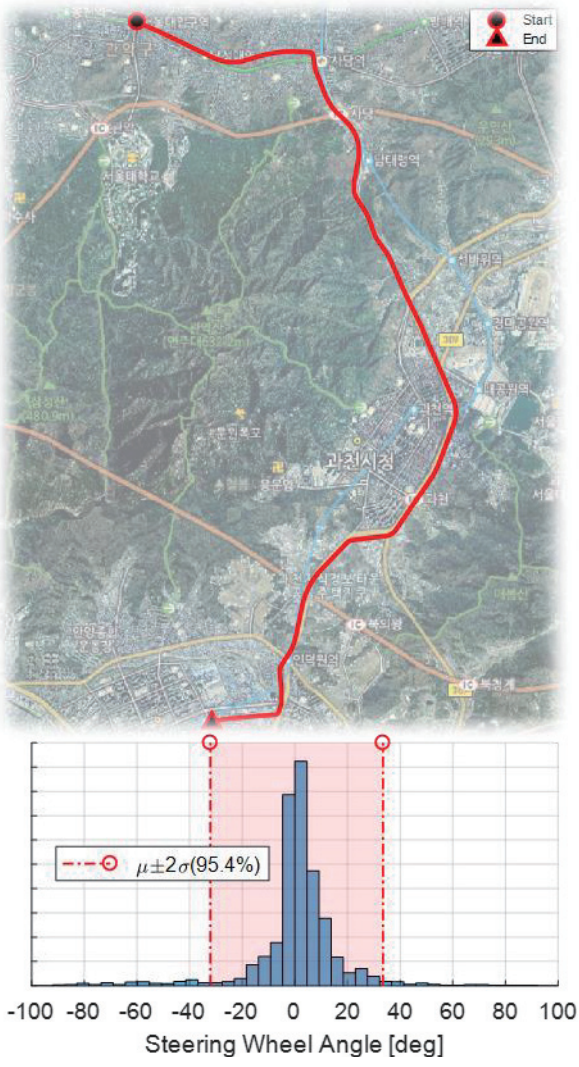

Figure 1.3. Simple Urban Driving Course and Steering Wheel Angle Usage Range

Certainly, the on-center steering wheel angle is mainly used for driving. However, it is also evident that the off-center steering wheel angle range is essential for our “last-mile” driving. U-turn is one of the typical scenarios that occur in a chaotic urban. A driver encounters situations frequently that need a large steering wheel angle in “last-mile” driving. Therefore, the steering control module in a fully-automated driving system has to be able to cover the challenging scenarios that require a full-range of steering wheel angle.

This dissertation proposes the steering controller, which is capable of tracking the desired path from straight to the minimum turning radius using the future driving trajectory prediction.

1.2. Previous Researches

The steering control is one of the long-standing topics in the vehicle control field. In the early stages of introducing an automated driving vehicle to the public, the automatically rotating of the steering wheel was considered to be a first point that distinguishes an automated driving vehicle from conventional vehicles.

Seamless tracking of the pre-defined path was considered to be the most basic function of an automated driving system. In 2017, Baidu initiated the Apollo project that makes public the element technologies for automated driving. The first released function was the basic vehicle control module, which tracks the pre-defined path and the speed profile. From the first release, the Apollo system has developed the additional functions for autonomous driving based on the steering control module.

There have been a number of approaches for the steering control. Most of the steering control techniques have been applied for the path-tracking problem. The path-tracking problem has been addressed as an error-regulation problem.

(Rajamani, 2011) introduced the steering controller, which makes a vehicle to stay in the centre of its lane. An error-dynamics for the lateral position error and the heading error was defined. The vehicle model, which was embedded in the error-dynamics, is a single-track model with a linear tire model. Under the small slip-angle and the constant vehicle speed assumption, the state-space model for the lateral error-dynamics was given as linear system dynamics. And

then, a linear quadratic regulator (LQR) approach was applied for the regulation of defined lateral error-dynamics.

Similarly, an error-state feedback control method has been utilized frequently. (Kritayakirana & Gerdes, 2012) designed a steering controller for an autonomous race car. The combination of feedforward and feedback steering controller has been applied to track the desired path. As the main contribution, (Kritayakirana & Gerdes, 2012) proposed the error-dynamics referenced to the centre of percussion(COP) from the desired path. The lateral force at the rear tire that affects the lateral dynamics of a vehicle can be negligible at the COP. The closed-loop system is proved to remain stable even when a vehicle is in limit-handling.

An error-state feedback control or LQR approach are also called an optimal control approach. The steering controller in the Apollo project also deploys the LQR method. The stability of LQR in control theory has already been proved well. However, the parameters that affect tracking performance are tuned indirectly. Finding control parameters for desirable tracking performance is a time-consuming job because the stability of the controller is not directly linked to the desirable tracking performance. Even if the tuning procedure has been done, the additional tuning process is probably needed when the characteristics of a target vehicle are changed.

The performance analysis for a linear system is a well-established topic in control theory. However, several assumptions are necessary to construct the error-dynamics in a linear form. From that point, a sliding mode control (SMC) approach has been applied for the path-tracking control problem as a

conventional method. The SMC approach is proven to be a practical and effective way to regulate non-linear system dynamics. The SMC shows fast response and robustness against system uncertainties and external disturbance with respect to the non-linear system. (Song, Hedrick, & Kang, 2014) and (Guo, Ge, Yang, & Li, 2014) successfully employed the SMC approach to the path-tracking problem.

As an alternative approach that is similar to the SMC but more effective, (Goh & Gerdes, 2016) and (Laurense, Goh, & Gerdes, 2017) proposed a hierarchical control structure. In the structure, the desired yaw-rate for the path-tracking is calculated by imposing stable 1^{st}-order dynamics to the non-linear lateral error states. And then, the steering control command is determined to track the desired yaw-rate with respect to a non-linear vehicle model. The calculation of the desired yaw-rate is similar to the synthetic input method-based controller design framework. (Gerdes & Hedrick, 1997; Sanketi, Zavala, & Hedrick, 2005; Subosits & Gerdes, 2017)

Besides the error regulation approach, the pure-pursuit method is one of the most popular and simplest approaches. The simplicity of the method is originated from that the pure-pursuit method is based on a geometric vehicle model. In the geometric path-tracking method, a vehicle model is simplified to an Ackermann steered single-track model. In the pure-pursuit method, it is assumed that the rear axle of a vehicle will travel along a constant radius circle at the given steering angle. The assumed vehicle motion is illustrated in Figure 1.4.

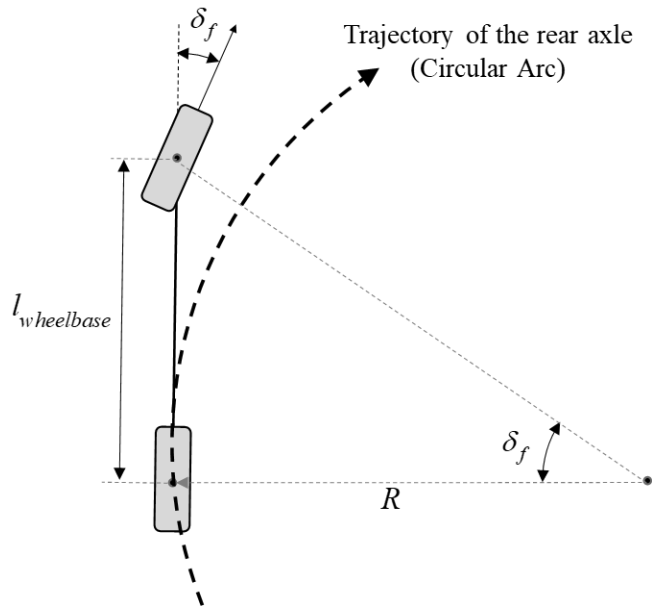

Figure 1.4. Geometrical representation: Turning behavior of an Ackermann steered single-track model

The turning radius of a trajectory that the rear axle travels along can be written with a wheelbase of the vehicle and a front steering angle as follow:

$$\tan(\delta_f) = \frac{l_{wheelbase}}{R} \tag{1.1}$$

where δ_f denotes the steering angle of a front wheel, $l_{wheelbase}$ denotes the wheelbase of the vehicle, R denotes the turning radius of the trajectory that the rear axle travels along at the given steering angle.

The Pure Pursuit method calculates a desired curvature of the vehicle trajectory based on the turning behavior of the Ackermann steered vehicle. The trajectory is a circular arc that crosses both the rear axle location and a target point located on the desired path. The geometrical representation, described

above is illustrated in Figure 1.5. The desired curvature for the desired path tracking can be written as follow:

$$\sin(\alpha) = \frac{l_{prev}}{2R_{des}} \tag{1.2}$$

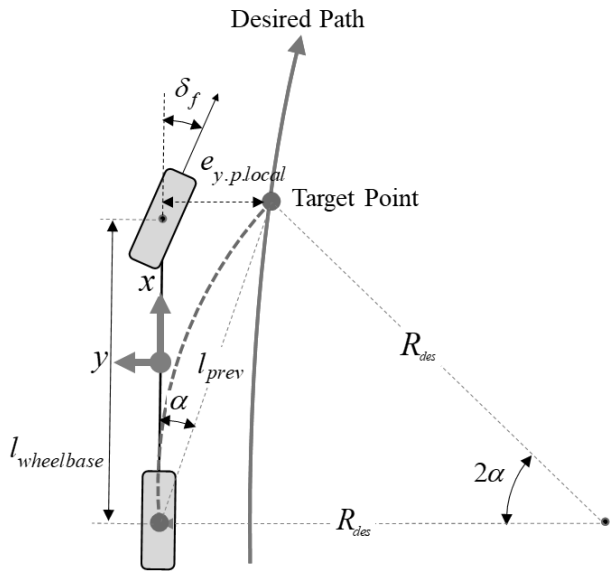

Figure 1.5. The desired curvature calculation for the path-tracking in the pure-pursuit method

Eq.(1.2) can also be rearranged as follow:

$$\sin(\alpha) = \frac{e_{y.p.local}}{l_{prev}} \tag{1.3}$$

From the Eq.(1.2) and Eq.(1.3), the desired curvature can be written with known values as follow:

$$\frac{1}{R_{des}} = \frac{2 \cdot e_{y.p.local}}{l_{prev}^2} \tag{1.4}$$

The desired steering wheel angle for the desired curvature realization can be written from the Ackermann steered vehicle model with the small angle assumption in Eq.(1.1).

$$\delta_{sw} = G_{str} \cdot \frac{2 \cdot l_{wheelbase} \cdot e_{y.p.local}}{l_{prev}^2} \tag{1.5}$$

where, G_{str} is a steering gear ratio and δ_{sw} is a steering wheel angle.

In the pure-pursuit method, the steering wheel angle input is proportional to the desired curvature, and the desired curvature is determined with the lateral error from the target point. The lateral error is calculated with respect to the vehicle coordinate system. Accordingly, the pure-pursuit method compensates the lateral path error indirectly. However, the performance of the pure-pursuit method is affected to the heading measurement and the steering actuation delay, because the lateral error which is directly included to the steering input, is calculated from the local coordinate system.

Another performance issue in the pure-pursuit method is the determination of the target point. Commonly, the faster a vehicle is, the preview length has to be longer. Therefore, the preview length, which determines the target point has to be proportional to the vehicle speed. In practical, the proportional gain to the vehicle speed and the lower bound are key parameters to determine the tracking performance.

Besides the error regulation approaches and the geometric approach, the

convex-optimization approaches have been spotlighted for the steering control problem due to versatile advantages. The Model Predictive Control (MPC) is one of the convex-optimization approaches. The MPC-based steering controller predicts the behavior of the lateral vehicle dynamics in the assumed time horizon and determines the optimal control input sequence for the minimization of a cost function. Under the optimization framework, an MPC-based steering controller can be designed to handle various driving scenarios, such as obstacle avoidance, lane change, as well as path-tracking (Falcone, Borrelli, Asgari, Tseng, & Hrovat, 2007; Funke, Brown, Erlien, & Gerdes, 2016; Suh, Chae, & Yi, 2018).

The convex-optimization technique can be employed to the motion planning because the optimal state sequences of system dynamics can also be obtained in real-time as well as the optimal input sequence (Subosits & Gerdes, 2019).

Although the MPC-based controller is versatile and widely used in the vehicle control community, several disadvantages exist inherently. The computation load is one of the challenging issues in the MPC-based controller because the real-time optimization procedure has to be done. The complexity of the system dynamics and additional constraints require more computation time exponentially. Even though the longer time horizon shows better performance normally, it also requires more time to compute.

Moreover, all the equations that construct the cost function and the constraints are regarded as mathematical conditions to the optimization solver, whereas an engineer attaches the physical meaning to the equations. Consequently, every variable in the optimization framework requires proper

scale factors. The selected scale factors mainly affect the performance of the controller.

1.3. Thesis Objectives

From a careful review of a considerable amount of literature, most of the researches for the automated driving system have mainly focused on the scenarios in a motorway or simple urban. However, a highly automated vehicle has to handle the complex scenarios in inner-city-street. Because the automated driving vehicle has to be capable of dealing with U-turn or mini roundabout scenarios in inner-city-street. Expanding of the automated driving scenarios implies expanding of the usage range of the steering wheel angle. In the motorway or simple urban, the on-center steering wheel angle range is mainly used, whereas the complex scenarios such as U-turn that occur in our "last-mile" driving, require a full range of steering wheel angle.

Although a number of the approaches have been proposed in the vehicle steering control problems, it seems that the further studies regarding the steering control approach which is able to operate within the lock-to-lock range are needed.

For that reason, this dissertation focused on developing the steering control algorithm, which is capable of tracking the desired trajectory from the near-straight to the minimum turning radius.

1.4. Thesis Outline

This dissertation is structured in the following manner. An overall architecture of the proposed automated driving vehicle control algorithm is described in Chapter 2. In Chapter 3, online vehicle motion learning method is described. The main objective of the online vehicle motion learning is to enhance applicability of the proposed steering controller to various vehicle type. In Chapter 4, the driving trajectory prediction procedure and the proposed steering controller is described. The predicted error in near-future is incorporated into the error dynamics. The proposed steering controller has a 2-layered architecture: Desired motion decision and desired motion tracking. The desired yaw-rate is determined as a synthetic input for the path-tracking. In order to track the desired yaw-rate, the desired steering wheel is determined using model reference adaptive control. Chapter 6 and Chapter 5 show the experiment and simulation results for the evaluation of the performance of the proposed algorithm. Then the conclusion which describes the summary and contribution of the proposed automated driving control algorithm and future works is presented in Chapter 7.

Chapter 2

An Overview of the Proposed Steering Control System for an Automated Driving Vehicle

Recently, there have been various attempts based on a machine learning approach, which suggests an end-to-end architecture of an automated driving system from the advancement in the computer science field (Bojarski et al., 2016). However, the modular architecture of an automated driving system has been more preferred in much research because the modular architecture is favorable to modification and troubleshooting.

A steering controller, which is proposed in this dissertation, was developed for an automated driving system with a modular architecture that consists of three parts: an environment representation, motion planning, and vehicle control. A vehicle pose and the desired path are inputs of the proposed steering

controller. The vehicle pose, which is estimated by a localization module in the environment representation part, contains a position and heading information. The motion planning module plans the desired path. Accordingly, the motion planning module carries out finding an optimal path that a vehicle can drive along safely, based on an understanding of the surrounding environment.

The proposed steering controller determines the desired steering angle, which is appropriate to track the desired path. The vehicle control system architecture, including the proposed steering controller, is outlined in Figure 2.1. The proposed steering controller consists of four sub-modules: the online vehicle motion learner, desired motion decision, desired motion tracker, and steering torque controller. The three sub-modules except for the steering torque controller are mainly focused on this dissertation.

The online vehicle motion learner identifies a vehicle model incrementally in real-time with the driving data stream. The vehicle model is represented using a Gaussian Process (GP) model. An Incremental Sparse Spectrum Gaussian Process Regression (ISSGPR) method has been applied for the online learning and identification of the vehicle model. The ISSGPR is an incremental version of the Gaussian Process Regression (GPR) method. A kernel function in the GP model is approximated using randomly drawn spectral frequencies in the ISSGPR.

In this research, the process that determines the desired steering wheel angle for the desired trajectory tracking is divided into two parts. The desired motion decision determines the desired yaw-rate of the vehicle for the desired trajectory

tracking, and then the desired motion tracker calculates the desired steering wheel angle for the desired yaw-rate tracking.

In most cases, the vehicle controller is designed with the incorporation of the vehicle dynamics into the error dynamics definition. The separation strategy in this research is for the exclusion of the time-consuming process that the modeling parameter identification. One of the main objectives of this research is the exclusion of prior knowledge about the vehicle model for the practicality and applicability.

The desired motion decision module compensates for the current error and the predicted future error both. In order to determine the predicted future error, the proposed algorithm predicts the future driving trajectory of a vehicle from the current states. The vehicle model information from the online vehicle motion learner is utilized to generate the future driving trajectory. The vehicle model predicts the yaw-rate motion from the current steering wheel angle and rotating speed. The future driving trajectory is similar to parking guidelines in the commercialized driver assistant system. The predicted future error acts as the penalty term for the excessive/deficient steering input. Accordingly, the proposed steering controller shows outstanding performance to track the trajectory from the straight course to the minimum turning radius course. That is one of the main contributions of this research.

The desired motion tracker module calculates the desired steering wheel angle to track the desired yaw-rate of the vehicle using the Model Reference Adaptive Control (MRAC) approach. It is unnecessary to obtain the vehicle dynamics modeling parameter because the MRAC approach approximates the

vehicle yaw-rate model as the 1st-order delay system with unknown parameters. The MRAC adapts the control gain set to make the unknown 1st-order delay system track the desired system trajectory. The desired system trajectory is the low-pass filtered desired yaw-rate.

The outstanding performance of the proposed steering controller in the tracking he arbitrary path has been validated via vehicle tests and simulations with challenging scenarios. Notably, the achievements have been made with no prior knowledge of the vehicle model parameter. It can be considered that the proposed steering controller achieves the practicality and applicability.

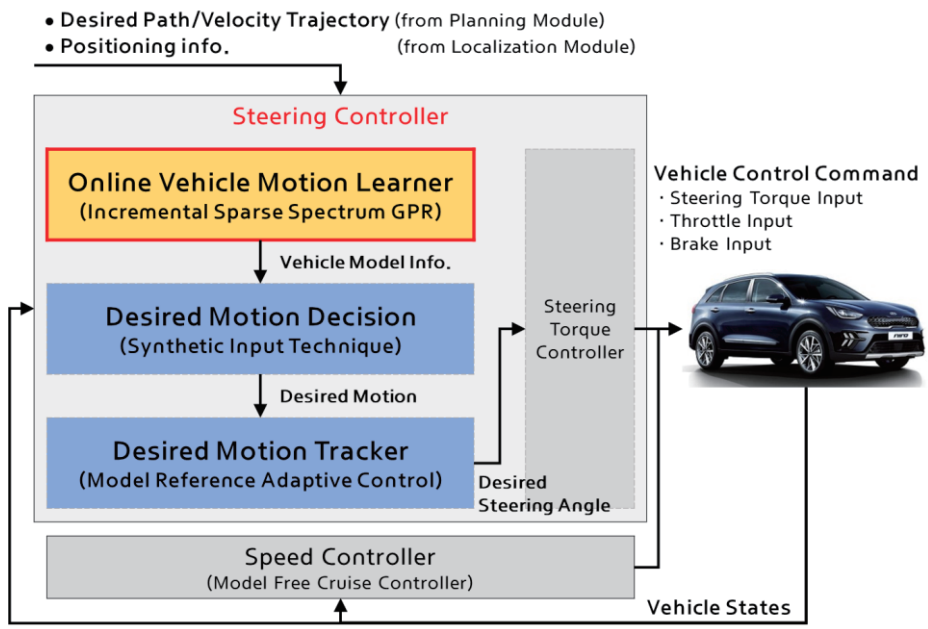

Figure 2.1. An Architecture of the Proposed Steering Control System for an Automated Driving Vehicle

Chapter 3

Online Vehicle Motion Learning

The predicted path error in the future is incorporated into the error dynamics for the desired motion decision. For the future path error prediction of the subject vehicle, the vehicle dynamic model has to be identified.

Various vehicle models have been proposed and employed for the vehicle control. A kinematic bicycle model describes the lateral motion of a vehicle without considering the external forces. The kinematic model is purely based on geometric relationship. The slip angle at both the front and rear wheel is assumed to be zero. The zero slip angle assumption is reasonable at low speeds because, the total lateral force from each tire varies quadratically with the vehicle speed for the lateral motion of the vehicle. Typically, it is known that the kinematic model is applicable for the vehicle speed less than $5 m/s$ (Rajamani, 2011)

A planar single-track model is the most popular model for the lateral vehicle control in tremendous previous research. There have been diverse variants that varies with the tire model.

However, the tire model parameters are difficult to be identified in practice, because the parameters are unmeasurable directly. Furthermore, the characteristics of the tire is time-varying due to wear, weather, friction coefficient of road, etc. Therefore, the parameters are considered as one of the design parameter in some cases.

Besides the tire model parameters, there are parameters, used as roughly approximated value, such as moment of inertia, and location of center of gravity, mass, etc. Even though numerous research has been conducted to suggest vehicle parameter estimation methods (Hong, Lee, Borrelli, & Hedrick, 2014; Lee, Hedrick, & Yi, 2004), the discrepancy in response between actual and model is covered by robustness of vehicle controller in most cases.

In order to identify the vehicle model parameter including tire model parameter, repeated simulations have to be conducted using the vehicle test data, such as double lane change maneuver. The simulation is repeated until the combination of the model parameters which represents the actual response from the vehicle test data is found. Although the vehicle model parameter identification procedure is quite time-consuming, it is inevitable procedure for vehicle controller design.

In this dissertation, reducing an amount of time that is required to apply the proposed steering controller is one of the main objectives. The vehicle model for the driving trajectory prediction is identified in real-time using the streaming data of the vehicle. In order for the identification, the vehicle model is defined as Nonlinear AutoRegressive eXogenous(NARX) model. After that, Incremental Sparse Spectrum Gaussian Process Regression(ISSGPR) method

is employed to identify the NARX model.

3.1. Lateral Vehicle Dynamics Modeling using Nonlinear AutoRegressive eXogenous Model

For the driving trajectory prediction in the near future, the vehicle model, which is different with the model, which is updated adaptively in MRAC is employed. In terms of vehicle control, discrepancy between actual response and model response is covered by robustness of controller. However, in terms of the prediction, the discrepancy is directly affected to the performance.

The vehicle dynamic model, which is intended to be identified, describes the relationship between the yaw-rate of the vehicle and the steering wheel angle. The proposed steering controller, which is based on the driving trajectory prediction, aims to cover the full range of usage of the steering wheel angle. Therefore, the vehicle model for the prediction has to represent the actual vehicle motion well in entire range of the steering wheel angle.

The yaw-rate response of the vehicle can vary with the steering wheel angular speed, the number of passengers, and the range of the steering wheel angle, etc. It is difficult to represent the actual vehicle response with the deterministic model in the all circumstance. And the current yaw-rate of the vehicle is dependent on previous series of yaw-rate and steering wheel angle. From the above characteristics of the system, the vehicle dynamics modeling can be

considered as time-series modeling based on a Nonlinear AutoRegressive model with eXogenous output (NARX).

The NARX model have been employed widely to forecast the behavior of the system. The application of the NARX model can vary from the dynamical system to stock market (Chan, Yuen, Lee, & Arashpour, 2015; M'ng & Mehralizadeh, 2016) .

(Chan et al., 2015) exploited the NARX model to predict the hysteretic behavior of passive control systems. (Chan et al., 2015) compared with the deterministic hysteresis models, the Bouc-Wen-Baber-Noori (BWBN) model, and it was shown that the NARX model performed better than the deterministic model method.

The proposed vehicle model in the form of NARX model can be stated as:

$$\hat{y}_{pred.k} = f_{pred}\left(y_{pred.k-1}, \quad u_{pred.k-1}, \quad u_{pred.k-2}\right) \tag{3.1}$$

$$= y_{pred.k-1} + T_{s.pred} \cdot g_{pred}\left(y_{pred.k-1}, \quad u_{pred.k-1}, \quad u_{pred.k-2}\right) \tag{3.2}$$

An output of the proposed vehicle model, Eq.(3.1), is defined as the yaw-rate of the vehicle:

$$y_{pred.k} = \gamma_k \tag{3.3}$$

An input of the proposed vehicle model, Eq.(3.1), is defined as the multiplication of the longitudinal velocity and the steering wheel angle.

$$u_{pred.k} = v_{x.k} \cdot \delta_k \tag{3.4}$$

The yaw-rate of the vehicle can vary with the longitudinal velocity under the consistent steering wheel angle. Therefore, the longitudinal velocity has to be

combined with the steering wheel angle. Especially, it is well-known that the yaw-rate of the vehicle is proportional to the product of the longitudinal velocity of the vehicle and the steering wheel angle in the Ackermann steer relation. The higher speed the vehicle gets, the more discrepancy between Ackermann steer relation and actual vehicle behavior occurs due to the nonlinearity. However, it is clear that the product of the velocity and the steering wheel angle is still a main component which generates the yaw-rate of the vehicle.

The NARX model, Eq.(3.1), is identified with Gaussian Process Regression(GPR) technique in 3.2. The GPR is one of the black-box modeling approach, and the output of the GPR model depends on the training data. In order to reduce the effect of the over-fitting problem, Eq.(3.1) is re-formulated as Eq.(3.2).

The function, $g_{pred}(\bullet)$, can be interpreted as the yaw acceleration of the vehicle in the Euler discretization method. It is clear that the difference between the current yaw-rate and the previous yaw-rate is bounded, because the yaw-rate difference is constrained to the vehicle dynamics. The mean function of $g_{pred}(\bullet)$, which is a prior knowledge of GPR model, is assumed to be zero. That means the current yaw-rate would be identical to the previous yaw-rate as the default. It can be a reasonable strategy when the data is not enough to identify the vehicle model.

3.2. Incremental Sparse Spectrum Gaussian Process Regression

In order to identify the vehicle model in the form of Eq.(3.2), the Gaussian Process Regression (GPR) method is employed. The GPR method is widely used for black-box modeling with input-output data pair. The GPR method is fueled by the data of the target system only and predicts the output for the query input point. However, the growth of the kernel expansion is the main disadvantage that makes it difficult to be used for the online application, because the size of the kernel matrix is limited, and the computation load of the inversion of the matrix is increased exponentially with its size. The data selection procedure that determines the data to be removed can be applied to maintain a fixed computation load on the kernel expansion. However, the data selection can cause the modeling result to be biased because the selection criteria are built on the intuition of an engineer. Therefore, the supplement of the data selection has to be considered carefully.

In order to avoid drawbacks above, the Incremental Sparse Spectrum Gaussian Process Regression (ISSGPR) method has been proposed (Gijsberts & Metta, 2013). The drawbacks of the kernel machinery can be resolved by the weight-space view representation of the dynamic system to be identified. The weight-space view representation assumes that the output of the dynamic system can be represented as a linear combination of the non-linear feature mapping and its weight vector. The non-linear feature mapping lifts the system

input vector to the higher dimension. The simplest feature mapping function is an identity function.

The non-linear feature mapping function, used in the ISSGPR method, has to be a vector that can represent the Gaussian kernel with the inner-product of itself. However, the Gaussian kernel, which is one of the Radial Basis Function (RBF) typed kernels, is infinite-dimensional. The sparsification technique that approximates the shift-invariant kernel functions with a finite-dimensional feature mapping function is applied to resolve this problem. A finite-dimensional feature mapping function is constructed by the randomly drawn spectral frequencies, and that way is similar to the Fourier transformation. This sparsification method was proposed for the large-scale batch learning problem originally (Rahimi & Recht, 2008a). (Gijsberts & Metta, 2013) deployed this technique with an incremental update rule, and made the GPR to be utilized in the online application with constant computational complexity. The performance of the ISSGPR method has been shown via experimental validation (Droniou, Ivaldi, Stalph, Butz, & Sigaud, 2012; Gijsberts & Metta, 2013).

3.2.1. The weight-space view of Gaussian process regression

Gaussian process regression(GPR) method is started from a training data set:

$$\Xi = \left\{ \left(\boldsymbol{x}_i, y_i \right) \middle| i = 1, \cdots, N \right\} \tag{3.5}$$

where, $y \in \mathbb{R}^{1\times 1}$, $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$ denote an output and an input of the model respectively. N is the number of points in the training data set.

The GPR makes inferences about the relationship between the inputs and the output from the given training set. The ultimate goal of the GPR model is to infer the predictive distribution with respect to given query input.

GPR model can be interpreted in two ways: the weight-space view, and the function-space view (Rasmussen, 2003).

In the weight-space view, the model output is assumed to be an inner-product between the weight vector and the nonlinear feature map. It is also assumed that the outputs are corrupted by independent identically distributed Gaussian noise. Accordingly, the model can be written as:

$$y = \boldsymbol{\phi}^T(\boldsymbol{x}) \cdot \boldsymbol{w} + \varepsilon \tag{3.6}$$

where $\varepsilon \sim N(0, \sigma_\varepsilon)$ is independent identically distributed Gaussian noise. $\boldsymbol{w} \in \mathbb{R}^{2D\times 1}$ is the weight vector. $\boldsymbol{\phi}(\boldsymbol{x}) : \mathbb{R}^{n\times 1} \to \mathbb{R}^{2D\times 1}$ is a nonlinear function which maps the n-dimensional input, $\boldsymbol{x}$ to the $2D$-dimensional feature space using a set of basis functions. The nonlinear feature mapping is utilized to overcome the limited expressiveness. If $\boldsymbol{\phi}(\boldsymbol{x})$ is an identity function, the output of the function model is limited to a linear combination of input elements itself.

Furthermore, infinite dimensional basis function is intractable, and the basis function has to be chosen by an engineer with intuition about the system to be modeled.

From the above noise assumption, the likelihood of the observed outputs, $\boldsymbol{y} = \left[y_1, \cdots, y_N\right]^T \in \mathbb{R}^{N\times 1}$ can be written with the given design matrix,

$\boldsymbol{\Phi} = \left[\boldsymbol{\phi}(\boldsymbol{x}_1), \cdots, \boldsymbol{\phi}(\boldsymbol{x}_N)\right] \in \mathbb{R}^{N \times 2D}$ as follow:

$$p\left(\boldsymbol{y} \middle| \boldsymbol{\Phi}, \boldsymbol{w}\right) = N\left(\boldsymbol{\Phi}\boldsymbol{w},\ \sigma_\varepsilon^2 \boldsymbol{I}_N\right) \tag{3.7}$$

The prior distribution over the weight vector can be defined with zero mean and covariance matrix $\Sigma_{\boldsymbol{w}}$ as follow:

$$p\left(\boldsymbol{w}\right) = N\left(\boldsymbol{0}, \Sigma_{\boldsymbol{w}}\right) \tag{3.8}$$

From the prior distribution over weight vector and the likelihood of the observed output, the posterior distribution over the weight vector can be derived based on Bayes' rule.

$$p\left(\boldsymbol{w} \middle| \boldsymbol{\Phi}, \boldsymbol{y}\right) = N\left(A^{-1}\boldsymbol{\Phi}^T \boldsymbol{y},\ \sigma_\varepsilon^2 A^{-1}\right) \tag{3.9}$$

where $A = \boldsymbol{\Phi}^T\boldsymbol{\Phi} + \sigma_\varepsilon^2 \Sigma_{\boldsymbol{w}}^{-1}$

Bayes' rule describes that the posterior distribution is proportional to the product between the likelihood distribution and the prior distribution as Eq.(3.10). The marginal likelihood is known as the normalizing constant, which is independent of the weights.

$$\underbrace{p\left(\boldsymbol{w} \middle| \boldsymbol{\Phi}, \boldsymbol{y}\right)}_{\text{Posterior}} = \frac{\overbrace{p\left(\boldsymbol{y} \middle| \boldsymbol{\Phi}, \boldsymbol{w}\right)}^{\text{Likelihood}} \cdot \overbrace{p\left(\boldsymbol{w}\right)}^{\text{Prior}}}{\underbrace{p\left(\boldsymbol{y} \middle| \boldsymbol{\Phi}\right)}_{\text{Marginal likelihood}}} \propto \underbrace{p\left(\boldsymbol{y} \middle| \boldsymbol{\Phi}, \boldsymbol{w}\right)}_{\text{Likelihood}} \cdot \underbrace{p\left(\boldsymbol{w}\right)}_{\text{Prior}} \tag{3.10}$$

Note that the expectation of the posterior distribution is called the maximum a posteriori (MAP) estimate of the weight vector.

In the Bayesian framework, the predictive distribution over the output with respect to a query input is obtained from Eq.(3.9).

$$p\left(y_q \middle| \boldsymbol{\phi}_q, \boldsymbol{\Phi}, \boldsymbol{y}\right) = N\left(\boldsymbol{\phi}_q^T A^{-1} \boldsymbol{\Phi}^T \boldsymbol{y},\ \sigma_\varepsilon^2 \boldsymbol{\phi}_q^T A^{-1} \boldsymbol{\phi}_q\right) \tag{3.11}$$

where $\boldsymbol{\phi}_q = \boldsymbol{\phi}\left(\boldsymbol{x}_q\right)$, y_q is the predicted output at a given query input, $\boldsymbol{x}_q$.

Note that the expectation of the posterior distribution implies the weight vector for the query input, $\boldsymbol{\phi}_q$.

3.2.2. The function-space view of Gaussian process regression

Gaussian process regression can also be interpreted in the function-space view. It will be shown that the function-space view is identical to the weight-space view.

In the function-space view, the model between the inputs and the outputs is written as:

$$y = f\left(\boldsymbol{x}\right) + \varepsilon \tag{3.12}$$

where $\varepsilon \sim N(0, \sigma_\varepsilon)$ is independent identically distributed Gaussian noise. An arbitrary function, f maps the input to the output.

Note that the function model is not specified concretely, thus GPR in the function-space view is one of the black-box modeling approach. Accordingly, it is difficult to divide into the likelihood and the prior distribution in contrast with the case in the weight-space view.

The inference in the function-space view is represented using a kernel (or covariance) function. The kernel function quantifies the similarity between the inputs. Various kernel function models can be applied. In this dissertation, Asymmetric Squared Exponential (ASE) kernel function is employed. The ASE kernel function is also known as the anisotropic Radial Basis Function (RBF) kernel. The ASE kernel is given by:

$$\forall i, j \in B$$

$$k_{ij} = \mathrm{cov}\left(y_i, y_j\right) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 e^{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T \boldsymbol{M}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)} \tag{3.13}$$

where, $B = \{1, \cdots, N\}$, k_{ij} denotes i th row and j th column element of the kernel matrix, $\boldsymbol{K}\left(\boldsymbol{X}, \boldsymbol{X}\right)$. $\boldsymbol{M} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $M_{ii} = l_i^{-2}$. l_i is called the characteristic length scale, which describes the relative importance of each input element and plays a role as overall bandwidth.

Note that the exponent term in Eq.(3.13) is the Euclidean distance square with a weighting matrix, $\boldsymbol{M}$, and it denotes the similarity between two inputs. The interesting characteristic of the kernel function is that covariance between two outputs is represented as a function of two correspond inputs. Therefore, the closer the Euclidean distance between two inputs is, the larger covariance value is calculated between the correspond two outputs.

From the above, the kernel matrix, $\boldsymbol{K}\left(\boldsymbol{X}, \boldsymbol{X}\right)$ can be written as:

$$\boldsymbol{K}\left(\boldsymbol{X}, \boldsymbol{X}\right) = \begin{bmatrix} k\left(\boldsymbol{x}_1, \boldsymbol{x}_1\right) & \cdots & k\left(\boldsymbol{x}_1, \boldsymbol{x}_N\right) \\ \vdots & \ddots & \vdots \\ k\left(\boldsymbol{x}_N, \boldsymbol{x}_1\right) & \cdots & k\left(\boldsymbol{x}_N, \boldsymbol{x}_N\right) \end{bmatrix} \in \mathbb{R}^{N \times N} \tag{3.14}$$

where, $\boldsymbol{X} = \left[\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\right]^T \in \mathbb{R}^{N \times n}$ is a design matrix which aggregates the input vector in the training data set, Ξ.

Note that the kernel matrix contains covariance values with respect to all input pairs in the training data set, Ξ. When a query input point is given, the kernel matrix is utilized to determine where the predicted output is located among the outputs in the training data set. Because how close the query input point is from the inputs in the training data can be calculated from the kernel matrix.

From the definition of covariance function, the prior knowledge of the function model can be defined as Eq.(3.15). The mean of the function is assumed to be zero vector.

$$f(\boldsymbol{X}) \sim \left(\boldsymbol{0}, \quad \boldsymbol{K}\left(\boldsymbol{X}, \boldsymbol{X}\right)\right) \tag{3.15}$$

The joint distribution between the observed outputs and the output at the query point under the prior as:

$$\begin{bmatrix} \boldsymbol{y} \\ f_q \end{bmatrix} \sim \left(\boldsymbol{0}, \quad \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{x}_q) \\ \boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) & k\left(\boldsymbol{x}_q, \boldsymbol{x}_q\right) \end{bmatrix} \right) \tag{3.16}$$

where, $\boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) = \left[k\left(\boldsymbol{x}_q, \boldsymbol{x}_1\right) \quad \cdots \quad k\left(\boldsymbol{x}_q, \boldsymbol{x}_N\right) \right] \in \mathbb{R}^{1 \times N}$, σ_ε^2 is added to the covariance function of the output distribution for the realistic modeling as follow:

$$\operatorname{cov}\left(y_i, y_j\right) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sigma_\varepsilon^2 \tag{3.17}$$

The predictive distribution over the output at the given query input point is

corresponding to conditioning the joint distribution Eq.(3.16)

$$p\left(f_q \middle| \boldsymbol{x}_q, \boldsymbol{X}, \boldsymbol{y}\right) = N\left(m_q, s_q\right) \tag{3.18}$$

$$m_q = \boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) \cdot \left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N\right]^{-1} \cdot \boldsymbol{y} \tag{3.19}$$

$$s_q = \boldsymbol{k}(\boldsymbol{x}_q, \boldsymbol{x}_q) + \boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) \cdot \left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N\right]^{-1} \cdot \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{x}_q) \tag{3.20}$$

In the following, it will be shown that the function-space view is identical to the weight-space view. It begins with the *kernel trick*. The ASE kernel, which is one of the radial basis function(RBF) can be represented in terms of inner-product in nonlinear features space. It can be proved in the following example with a simple RBF kernel case as follow:

$$\boldsymbol{k}_{simple}\left(v_i, v_j\right) = \mathrm{e}^{-\left(v_i - v_j\right)^2} \tag{3.21}$$

$$= \mathrm{e}^{-v_i^2} \cdot \mathrm{e}^{-v_j^2} \cdot \mathrm{e}^{2v_i v_j} \tag{3.22}$$

$$= \mathrm{e}^{-v_i^2} \cdot \mathrm{e}^{-v_j^2} \cdot \underbrace{\sum_{m=0}^{\infty} \frac{2^m \cdot v_i^m \cdot v_j^m}{m!}}_{e^{2v_i v_j}} \tag{3.23}$$

$$= \sum_{m=0}^{\infty} \left\{ \underbrace{\left(\mathrm{e}^{-v_i^2} \cdot v_i^m\right)}_{\phi_m(v_i)} \cdot \underbrace{\frac{2^m}{m!}}_{C_m} \cdot \underbrace{\left(\mathrm{e}^{-v_j^2} \cdot v_j^m\right)}_{\phi_m(v_j)} \right\} \tag{3.24}$$

$$= \sum_{m=0}^{\infty} \left\{ \phi_m(v_i) \cdot C_m \cdot \phi_m(v_j) \right\} \tag{3.25}$$

$$= \phi^T(v_i) \cdot diag\left(\left[C_0, \cdots\right]\right) \cdot \phi(v_j) \tag{3.26}$$

From the above proof, it is shown that the RBF kernel can be represented as

the inner product of the nonlinear feature mapping, and the nonlinear feature map for the RBF kernel is infinite dimensional.

Without loss of generality, the ASE kernel can also be represented as the inner product of the infinite dimensional nonlinear feature map, $\boldsymbol{\phi}$.

$$\boldsymbol{k}(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \boldsymbol{\phi}^T(\boldsymbol{x}_i)\Sigma_{\boldsymbol{w}}\boldsymbol{\phi}(\boldsymbol{x}_j) \tag{3.27}$$

From the Eq.(3.27),

$$\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T \tag{3.28}$$

$$\boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) = \boldsymbol{\phi}_q^T\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T \tag{3.29}$$

From the Eq.(3.28), and Eq.(3.29), the mean of the predictive distribution of GPR in function-space view, Eq.(3.19) can be re-written as follow:

$$\begin{aligned} m_q &= \boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) \cdot \left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N\right]^{-1} \cdot \boldsymbol{y} \\ &= \left(\boldsymbol{\phi}_q^T \Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T\right) \cdot \left(\boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \boldsymbol{I}_N\right)^{-1} \cdot \boldsymbol{y} \end{aligned} \tag{3.30}$$

Let us begin with the following term:

$$\boldsymbol{\Phi}^T \cdot \left(\boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \boldsymbol{I}_N\right) \tag{3.31}$$

There are two ways in transforming Eq.(3.31):

$$\boldsymbol{\Phi}^T \cdot \left(\boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \boldsymbol{I}_N\right)$$

$$= \boldsymbol{\Phi}^T\boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2\boldsymbol{\Phi}^T \tag{3.32}$$

$$= \underbrace{\left(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \sigma_\varepsilon^2\Sigma_{\boldsymbol{w}}^{-1}\right)}_{A} \cdot \Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T \tag{3.33}$$

$$\boldsymbol{\Phi}^T \cdot \left(\boldsymbol{\Phi}\Sigma_{\boldsymbol{w}}\boldsymbol{\Phi}^T + \sigma_\varepsilon^2 \boldsymbol{I}_N\right)$$

$$= \boldsymbol{\Phi}^T \cdot \left(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N\right) \tag{3.35}$$

$$= A \cdot \Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T \qquad (3.34)$$

Therefore, Eq.(3.34) is identical to Eq.(3.35) as follow:

$$A \cdot \Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T = \boldsymbol{\Phi}^T \cdot \left(\boldsymbol{K} + \sigma_\varepsilon^2 \boldsymbol{I}_N \right) \qquad (3.36)$$

Multiplying the inverse of A to the left-side, and the inverse of $\left(\boldsymbol{K} + \sigma_\varepsilon^2 \boldsymbol{I}_N \right)$ to the right-side of Eq.(3.36) as follow:

$$\Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T \cdot \left(\boldsymbol{K} + \sigma_\varepsilon^2 \boldsymbol{I}_N \right)^{-1} = A^{-1} \boldsymbol{\Phi}^T \qquad (3.37)$$

In sequence, multiplying $\boldsymbol{\phi}_q^T$ to the left-side and $\boldsymbol{y}$ to the right-side of Eq.(3.37) as follow:

$$\boldsymbol{\phi}_q^T \Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T \cdot \left(\underbrace{\boldsymbol{K}\left(\boldsymbol{X}, \boldsymbol{X}\right)}_{\boldsymbol{\Phi} \Sigma_{\boldsymbol{w}} \boldsymbol{\Phi}^T} + \sigma_\varepsilon^2 \boldsymbol{I}_N \right)^{-1} \boldsymbol{y} = \boldsymbol{\phi}_q^T A^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} \qquad (3.38)$$

As shown in Eq.(3.11), the mean of the predictive distribution in the weight-space view is the right-side of Eq.(3.38). The left-side of Eq.(3.38) is the mean of the predictive distribution in the function-space view as shown in Eq.(3.30).

With similar way, it can be proved that the predictive variances in both weight-space and function-space views are identical.

In summary, Gaussian process regression can be interpreted in two ways, and the results from the both ways are equal.

The prior knowledge of the function model is assumed to be a zero mean function as Eq.(3.15). However, if the expected behavior of the system at the

given query input can be defined, it can be helpful to improve interpretability of the model with relatively small amount of training data.

The prior knowledge of the function with a deterministic mean function can be incorporated into the predictive distribution, Eq.(3.19), and then the mean of the predictive distribution is re-written as follow:

$$m_q = \boldsymbol{m}\left(\boldsymbol{x}_q\right) + \boldsymbol{K}(\boldsymbol{x}_q, \boldsymbol{X}) \cdot \left[\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_\varepsilon^2 \boldsymbol{I}_N\right]^{-1} \cdot \left(\boldsymbol{y} - \boldsymbol{m}\left(\boldsymbol{X}\right)\right) \tag{3.39}$$

where, $\boldsymbol{m}\left(\boldsymbol{x}_q\right) \in \mathbb{R}^{1\times 1}$, $\boldsymbol{m}\left(\boldsymbol{X}\right) \in \mathbb{R}^{N\times 1}$ are the deterministic mean function with respect to the query input and the inputs in training data set respectively.

Note that the predictive variance, s_q remains unchanged as Eq.(3.20), because the kernel matrix in the prior knowledge is only the function of the inputs.

3.2.3. Incremental Sparse Spectrum Gaussian Process Regression

The online vehicle model identification is one of the main contribution of this dissertation. As mentioned above, the vehicle dynamic system is time-varying and nonlinear system to be modeled analytically. Furthermore, the automated driving vehicle has to be applied to unstructured and non-stationary environment. Thus, the automated driving vehicle is required to learn from experience and adapt autonomously to changing condition with minimal human intervention.

In order to identify and update the vehicle model incrementally, Incremental Sparse Spectrum Gaussian Process Regression(ISSGPR) is proposed by

(Gijsberts & Metta, 2013).

There are some reasons that GPR is difficult to be implemented in real-time. The essential part of GPR, which makes black-box modeling enable with input-output data of the target system only, is kernel function. The kernel function calculation has to be done with respect to all combinations of input pairs in the training set in order to build the kernel matrix, $\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})$. The kernel matrix means the quantified experience from the training data. In real environment that input-output data of the system is streamed in real-time, the current query input point on the current time step becomes past experience in next time step. Preserving the training data set, and adding up the new data simultaneously means infinite expansion of the kernel matrix with exponentially increasing of computation load. Moreover, the kernel matrix has to be inverted to obtain the predictive distribution. Therefore, GPR with the kernel matrix directly is unable to be implemented in real-time.

In contrast with the function-space view, the linear model with the nonlinear feature mapping in the weight-space view allows exact updates of the predictive mean and variance. The derivation of incremental update rule is started with decomposition of predictive mean and variance, Eq.(3.11). The design matrix, $\boldsymbol{\Phi}$, and the stacked outputs, $\boldsymbol{y}$ in the current time step, k can be decomposed into previous time step term and the current time step term as follow:

$$\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{\Phi}_{k-1}^T & \boldsymbol{\phi}_k \end{bmatrix}^T \tag{3.40}$$

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{y}_{k-1}^T & y_k \end{bmatrix}^T \tag{3.41}$$

From the above decomposition, the matrix A in the current time step, k can be formulated as a recursive form:

$$A_k \equiv \boldsymbol{\Phi}_k{}^T \boldsymbol{\Phi}_k + \sigma_\varepsilon^2 \Sigma_{\boldsymbol{w}}^{-1} \tag{3.42}$$

$$= \left(\boldsymbol{\Phi}_{k-1}{}^T \cdot \boldsymbol{\Phi}_{k-1} + \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k^T \right) + \sigma_\varepsilon^2 \Sigma_{\boldsymbol{w}}^{-1} \tag{3.43}$$

$$= \underbrace{\left(\boldsymbol{\Phi}_{k-1}{}^T \cdot \boldsymbol{\Phi}_{k-1} + \sigma_\varepsilon^2 \Sigma_{\boldsymbol{w}}^{-1} \right)}_{\equiv A_{k-1}} + \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k^T \tag{3.44}$$

$$= A_{k-1} + \boldsymbol{\phi}_k \cdot \boldsymbol{\phi}_k^T \tag{3.45}$$

Let us define, $\boldsymbol{b}$ in the current time step, k and obtain a recursive form as follow:

$$\boldsymbol{b}_k \equiv \boldsymbol{\Phi}_k^T \boldsymbol{y}_k \tag{3.46}$$

$$= \underbrace{\boldsymbol{\Phi}_{k-1}^T \boldsymbol{y}_{k-1}}_{\equiv \boldsymbol{b}_{k-1}} + \boldsymbol{\phi}_k \cdot y_k \tag{3.47}$$

$$= \boldsymbol{b}_{k-1} + \boldsymbol{\phi}_k \cdot y_k \tag{3.48}$$

Note that the predictive mean and variance can be updated recursively from the updated A_k, and b_k in the current time step. However, an important issue remains here. As mentioned earlier, it has to be determined that what type of basis function for the nonlinear feature map is the best to model the system.

Therefore, it is time to retrieve the main idea of the function-space view regarding the kernel trick. As shown in the simple RBF case, the ASE kernel is represented by an infinite dimensional nonlinear feature mapping function. In order to utilize the above update rule, Eq.(3.45), and Eq.(3.48) with the

advantage of the kernel trick, the ASE kernel function has to be represented by a finite dimensional feature mapping function.

(Rahimi & Recht, 2008b) showed that the RBF kernel can be approximated to an arbitrary precision using a finite dimensional random feature mapping based on Bochner's theorem (Bochner, 1933).

As a result, the ASE kernel function, Eq.(3.13) is approximated by a finite-dimensional random feature as follow:

$$\forall i, j \in B$$
$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 e^{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T \boldsymbol{M}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)} \approx \boldsymbol{\phi}^T\left(\boldsymbol{x}_i\right) \cdot \boldsymbol{\phi}\left(\boldsymbol{x}_j\right) \tag{3.49}$$

$$\boldsymbol{\phi}(\boldsymbol{x}) = \frac{\sigma_f}{\sqrt{D}}\Big[\sin\left(\boldsymbol{\omega}_1^T \boldsymbol{x}\right), \cdots, \sin\left(\boldsymbol{\omega}_D^T \boldsymbol{x}\right), \cos\left(\boldsymbol{\omega}_1^T \boldsymbol{x}\right), \cdots, \cos\left(\boldsymbol{\omega}_D^T \boldsymbol{x}\right)\Big]^T \in \mathbb{R}^{2D \times 1} \tag{3.50}$$

where, $\forall i_\omega \in \{1, \cdots, D\}$, $\boldsymbol{\omega}_{i_\omega} \sim \left(\mathbf{0}_{n \times 1}, \boldsymbol{M}\right)$, and $\boldsymbol{\omega}_{i_\omega}$ is a randomly drawn spectral frequency vector.

Note that the above function approximation method is similar with Fourier transform. The recursive update rule, which is defined in Eq.(3.45), and Eq. (3.48) is completed with the finite-dimensional random feature mapping function.

Chapter 4

The Proposed Steering Control using Predicted Future Driving Trajectory

As mentioned before, the proposed steering controller has been developed to track the arbitrary trajectory in the automated driving system.

The proposed steering controller compensates for the predicted and the current error both. The predicted error is calculated from the predicted future driving trajectory. In a conventional way, the concept of the preview error has been used widely to design the steering controller. The preview error is calculated under the fixed current heading angle assumption. Consequently, the preview error is proportional to the preview distance or the velocity of a vehicle. Figure 4.1 illustrates the comparison between the predicted future error and the preview error definition in previous researches.

However, the heading of the vehicle varies constantly because the continuously varying steering input is applied to the vehicle. The proposed steering controller deploys the predicted the future driving trajectory, and the

predicted error is defined at the endpoint of the predicted driving trajectory. The future driving trajectory is similar to parking guidelines in the commercialized driver assistant system. The predicted future error acts as the penalty term for the excessive/deficient steering input.

The tracking error, which is defined with the incorporation of predicted error, is proposed, and the desired yaw-rate for regulating the tracking error is demonstrated. Finally, the desired steering wheel angle to track the desired yaw-rate without prior knowledge of the target vehicle dynamics is explained.

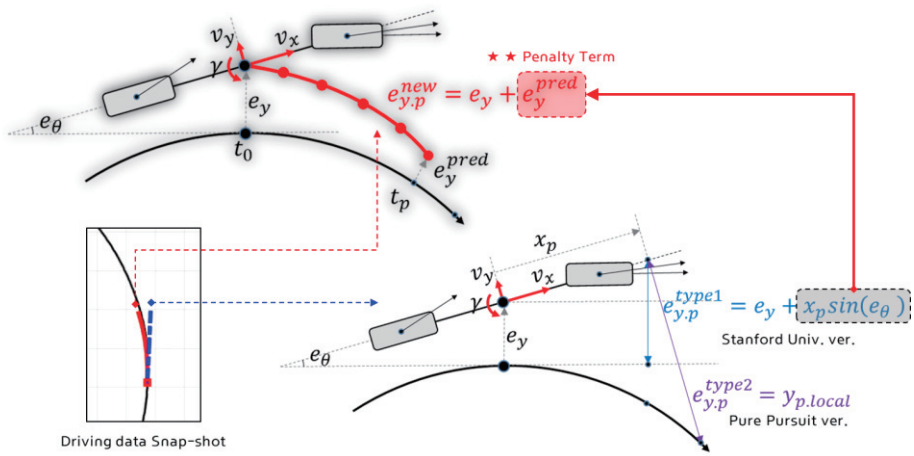

Figure 4.1. The comparison between the predicted future error and the preview error definition in previous researches

4.1. Future Driving Trajectory Prediction

From the identified and updated vehicle model in real-time, the driving trajectory is predicted. The driving trajectory is assumed to be the Euler spiral. The Euler spiral is commonly called various names, such as the clothoid or cornu-spiral. The Euler spiral has been widely used for a transition curve in highway road design. In automated driving technology, (Funke & Gerdes, 2016; Kapania, Subosits, & Gerdes, 2016; Talvala, Kritayakirana, & Gerdes, 2011; Theodosis & Gerdes, 2011) utilized the Euler spiral curve for local path planning in real-time.

Besides the Euler spiral, there have been various types of curve profiles, which are employed for the approximation of the vehicle driving trajectory.

As mentioned before, the future driving trajectory is predicted using the online learned vehicle model, which is described in Chapter 3. The vehicle model is defined as:

$$\hat{\gamma}_{pred.k} = f_{pred}\left(\hat{\gamma}_{pred.k-1},\quad \delta_{pred.k-1}\cdot v_{pred.k-1},\quad \delta_{pred.k-2}\cdot v_{pred.k-2}\right) \tag{4.1}$$

The Euler spiral based driving trajectory is obtained from the Fresnel (or Euler) integral as follow:

$$p_{X.p.s_f} = p_{X.k} + \int_{s_i}^{s_f} \cos\left(\psi_p(s)\right) ds \tag{4.2}$$

$$p_{Y.p.s_f} = p_{Y.k} + \int_{s_i}^{s_f} \sin\left(\psi_p\left(s\right)\right) ds \tag{4.3}$$

where, $\psi_p\left(s\right)$ is a predicted heading of the vehicle, which is a function of curvilinear abscissa, s .

The predicted heading profile is obtained from the predicted yaw-rate of the vehicle. The predicted yaw-rate is obtained from the vehicle dynamics model, which is identified online via the proposed method in this dissertation.

The yaw-rate prediction is conducted at every time step from the current time step to the prediction time horizon. The longitudinal velocity profile is given as the inputs for the prediction by a numerical integration. The longitudinal acceleration is assumed to be a constant as the longitudinal acceleration in the current time step. Similarly, the steering wheel angular rate is assumed to be a constant as the current steering wheel angular rate. The integral time horizon is discretized with a step size, $T_s = 0.1\text{sec}$.

$$\begin{aligned} &\forall i_p \in \boldsymbol{T} \\ &t_{profile} = \left\{ t_{i_p} \middle| t_{i_p} = i_p \cdot T_s \right\} \end{aligned} \tag{4.4}$$

$$v_{x.profile} = \left\{ v_{x.i_p} \middle| v_{x.i_p} = v_{x.k} + a_{x.k} \cdot t_{i_p} \right\} \tag{4.5}$$

$$\delta_{p.profile} = \left\{ \delta_{i_p} \middle| \delta_{i_p} = \delta_k + \dot{\delta}_k \cdot t_{i_p} \right\} \tag{4.6}$$

where, $\boldsymbol{T} \equiv \left\{0, 1, \cdots, n_p\right\}$, n_{pred} is the number of prediction step, and subscript $_k$ denotes the state, which is in the current time step. a_x denote the

longitudinal velocity and acceleration respectively. $\dot{\delta}$ denotes the steering wheel angular rate.

The Kalman filter is used to estimate the steering wheel angular rate. For the estimation, the state and its time invariant linear system structures can be defined as follow:

Process model (Continuous system)

$$\boldsymbol{x}_{str}(t) \equiv \begin{bmatrix} \delta(t) & \dot{\delta}(t) \end{bmatrix}^T \in \mathbb{R}^{2\times 1} \tag{4.7}$$

$$\begin{aligned} \dot{\boldsymbol{x}}_{str}(t) &= \boldsymbol{A}_{str} \cdot \boldsymbol{x}_{str}(t) + \boldsymbol{q}_{str} \\ &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \delta(t) \\ \dot{\delta}(t) \end{bmatrix} + \boldsymbol{q}_{str} \end{aligned} \tag{4.8}$$

$$\boldsymbol{q}_{str} \sim \left(\boldsymbol{0}_{2\times 1}, \quad \boldsymbol{Q}_{str} \right) \tag{4.9}$$

Measurement model

$$\begin{aligned} \boldsymbol{z}_{str.k} &= \boldsymbol{H}_{str} \cdot \boldsymbol{x}_{str.k} + \boldsymbol{v}_{str} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \delta_k \\ \dot{\delta}_k \end{bmatrix} + \boldsymbol{v}_{str} \end{aligned} \tag{4.10}$$

$$\boldsymbol{v}_{str} \sim \left(0, \quad \boldsymbol{V}_{str} \right) \tag{4.11}$$

where, $\boldsymbol{q}_{str} \in \mathbb{R}^{2\times 1}$ and $\boldsymbol{v}_{str} \in \mathbb{R}^{1\times 1}$ are noise process, which is assumed to be white, zero-mean, uncorrelated, and have known covariance matrices $\boldsymbol{Q}_{str}$ and $\boldsymbol{V}_{str}$ respectively.

The above process model, Eq.(4.8) can be discretized as follows:

Process model (Discrete system)

$$\begin{aligned} \boldsymbol{x}_{str.k} &= \underbrace{\left(\boldsymbol{I}_{2\times 2} + \boldsymbol{A}_{str} \cdot T_s\right)}_{\boldsymbol{F}_{str}} \cdot \boldsymbol{x}_{str.k-1} + \underbrace{\boldsymbol{q}_{str} \cdot T_s}_{\boldsymbol{w}_{str}} \\ &= \boldsymbol{F}_{str} \cdot \boldsymbol{x}_{str.k-1} + \boldsymbol{w}_{str} \end{aligned} \tag{4.12}$$

$$\boldsymbol{w}_{str} \sim \left(\boldsymbol{0}_{2\times 1}, \quad \boldsymbol{W}_{str}\right) \tag{4.13}$$

With above process and measurement model, the steering wheel angular rate is recursively estimated by the Kalman filter. A sequence of time and measurement update steps as follow:

Time update

$$\hat{\boldsymbol{x}}_{str.k}^{-} = \boldsymbol{F}_{str} \cdot \hat{\boldsymbol{x}}_{str.k-1}^{+} \tag{4.14}$$

$$\boldsymbol{P}_{str.k}^{-} = \boldsymbol{F}_{str} \cdot \boldsymbol{P}_{str.k-1}^{+} \cdot \boldsymbol{F}_{str}^{T} + \boldsymbol{W}_{str} \tag{4.15}$$

Measurement update

$$\boldsymbol{K}_{str.k} = \boldsymbol{P}_{str.k}^{-} \cdot \boldsymbol{H}_{str}^{T} \cdot \left(\boldsymbol{H}_{str} \cdot \boldsymbol{P}_{str.k}^{-} \cdot \boldsymbol{H}_{str}^{T} + \boldsymbol{V}_{str}\right)^{-1} \tag{4.16}$$

$$\hat{\boldsymbol{x}}_{str.k}^{+} = \hat{\boldsymbol{x}}_{str.k}^{-} + \boldsymbol{K}_{str.k} \cdot \left(\boldsymbol{z}_{str.k} - \boldsymbol{H}_{str} \cdot \hat{\boldsymbol{x}}_{str.k}^{-}\right) \tag{4.17}$$

$$\boldsymbol{P}_{str.k}^{+} = \left(I - \boldsymbol{K}_{str.k} \cdot \boldsymbol{H}_{str}\right) \cdot \boldsymbol{P}_{str.k}^{-} \cdot \left(I - \boldsymbol{K}_{str.k} \cdot \boldsymbol{H}_{str}\right)^{T} + \boldsymbol{K}_{str.k} \cdot \boldsymbol{V}_{str} \cdot \boldsymbol{K}_{str.k}^{T} \tag{4.18}$$

The predicted yaw-rate profile is obtained by the following as follow:

Algorithm 1

Input

Predicted velocity profile, Eq.(4.5) and steering wheel angle profile, Eq.(4.6)

For prediction time step, $k = 1, \cdots, N_p$

- If $k = 1$, Initialize the predicted states as the current states of the vehicle
- ELSE, Generate the predicted yaw-rate recursively by Eq.(4.1)

$$\hat{\gamma}_{pred.k} = f_{pred}\left(\hat{\gamma}_{pred.k-1},\quad \delta_{pred.k-1}\cdot v_{pred.k-1},\quad \delta_{pred.k-2}\cdot v_{pred.k-2}\right)$$

End

From the above procedure, the predicted yaw-rate profile is written as follow:

$$\gamma_{p.profile} = \left\{\gamma_0, \ldots, \gamma_{n_p}\right\} \tag{4.19}$$

From Eq.(4.19) the predicted heading profile can be generated by a numerical integration. The trapezoidal method is employed as a numerical integration method.

$$\forall i_p \in \boldsymbol{T}$$
$$\begin{cases} \psi_{i_p} = \psi_k & i_p = 0 \\ \psi_{i_p} = \psi_k + \dfrac{T_s}{2}\displaystyle\sum_{h=0}^{i_p-1}\left(\gamma_h + \gamma_{h+1}\right) & 1 \le i_p \le n_p \end{cases} \tag{4.20}$$

Note that the heading profile from Eq.(4.20) is defined in time domain. Therefore, Fresnel integral in Eq.(4.2), and Eq.(4.3) has to be done in time domain. The predicted driving trajectory is obtained using the trapezoidal numerical integration as follow:

$$\forall i_p \in \boldsymbol{T}$$

$$\begin{cases} p_{X.p.i_p} = p_{X.k} & i_p = 0 \\ p_{X.p.i_p} = p_{X.k} + \dfrac{T_s}{2}\sum_{h=0}^{i_p-1}\left\{v_{x.h}\cdot\cos\left(\psi_h\right) + v_{x.h+1}\cdot\cos\left(\psi_{h+1}\right)\right\} & 1 \le i_p \le n_p \end{cases}$$

(4.21)

$$\begin{cases} p_{Y.p.i_p} = p_{Y.k} & i_p = 0 \\ p_{Y.p.i_p} = p_{Y.k} + \dfrac{T_s}{2}\sum_{h=0}^{i_p-1}\left\{v_{x.h}\cdot\sin\left(\psi_h\right) + v_{x.h+1}\cdot\sin\left(\psi_{h+1}\right)\right\} & 1 \le i_p \le n_p \end{cases}$$

(4.22)

The driving trajectory prediction from the above method, is illustrated in Figure 4.2. The predicted future trajectory based on the proposed method is presented in a solid red line. The predicted trajectory with the assumption that the current heading of a vehicle is maintained is illustrated in a dash-dotted blue line. A solid black line is the actual driving trajectory of a vehicle.

As mentioned before, the preview model-based steering controller calculates the excessive desired steering wheel angle at the trajectory with the severe curvature. The severer the curvature of the desired trajectory is, the more unrealistic preview error is calculated. That characteristic of the preview model-based steering controller is handled with the adjustment of the preview distance. It can be challenging to find the tuning parameter, which can be applied to the general scenarios.

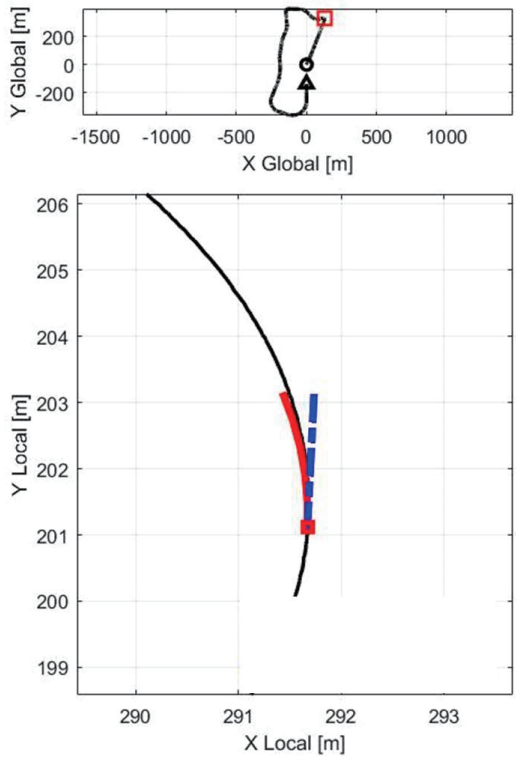

Figure 4.2. Driving trajectory prediction: The proposed prediction method vs The constant heading(preview) model

4.2. Tracking Error Definition and Calculation

In order to track the desired path, the proposed steering controller regulates the tracking error states. The tracking error states are defined from the relationship between a planar single-track vehicle model and the desired path, as shown in Figure 4.3

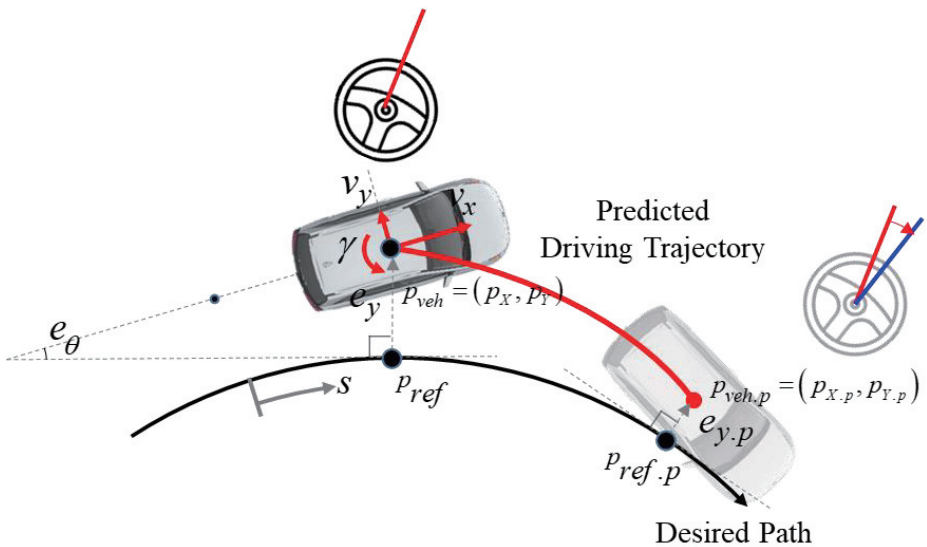

Figure 4.3. The illustration of the current and the predicted tracking error

where e_y is the lateral distance at the closest point on the desired path to the center of gravity (CG) of the vehicle model, p_{veh}. When p_{veh} is located at left-side of the desired path, a sign of e_y is defined to be positive. $e_{y.p}$ is the predicted lateral error in the future. The predicted lateral error is calculated

based on the predicted future driving trajectory. The detailed description of the driving trajectory prediction will be described in following chapters. In Figure 4.3, v_y, v_x, γ are lateral velocity, longitudinal velocity, and yaw-rate of the planar single-track vehicle model respectively. e_θ is heading error, which is defined to be the heading difference between the heading of the vehicle and the direction of the tangential vector at the reference point. The reference point, p_{ref}, is the closest point on the desired path from the CG point. The heading error can be written as follow:

$$e_\theta = \psi - \psi_d \tag{4.23}$$

where ψ is heading angle of a vehicle in the global coordinate. ψ_d is the direction of the tangential vector at the closest point on the desired path from the CG point of the vehicle.

From the above, the error dynamics system for the path-tracking is governed by the nonlinear equations of motion as below:

$$\dot{e}_y = v_x \sin(e_\theta) \tag{4.24}$$

$$\ddot{e}_y = \dot{v}_x \sin(e_\theta) + v_x \cos(e_\theta) \cdot \dot{e}_\theta \tag{4.25}$$

$$\dot{e}_\theta = \gamma - \kappa \cdot \dot{s} \tag{4.26}$$

$$\dot{s} \approx v_x \cos(e_\theta) \tag{4.27}$$

where κ is the curvature of the desired path at the reference point, P_{ref}. The derivative of curvilinear abscissa, $\dot{s}$ is defined to be a tangential velocity with respect to the reference point on the desired path as Eq.(4.27)

In most cases, the lateral velocity is unable to access in mass-produced vehicles. Although there have been tremendous research regarding the estimation of the lateral velocity, the lateral velocity is still difficult to be estimated with high-accuracy. However, in order to make passengers feel safe and comfortable, an automated vehicle has to drive smoothly. From the assumption that the vehicle drives safely and smoothly, the lateral velocity-related term is negligible in Eq.(4.24), Eq.(4.25), and Eq.(4.27). In previous research, the lateral velocity is often excluded in the definition of error dynamics. (Laurense et al., 2017) proposed a path-tracking controller at limit handling with the exclusion of the lateral velocity in error states definition.

The desired path is updated in the motion planning module at every time step. Accordingly, the reference point, P_{ref} on the desired path has to be determined at every time step so as to calculate the path-tracking error states. In our automated driving system, the desired path is forwarded in the form of several XY coordinates points to the vehicle control module. The desired path is approximated as a 4th-order polynomial curve from the coordinates as follow:

$$x_{des}(s) = C_{x.4} \cdot s^4 + C_{x.3} \cdot s^3 + C_{x.2} \cdot s^2 + C_{x.1} \cdot s^1 + C_{x.0} \tag{4.28}$$

$$y_{des}(s) = C_{y.4} \cdot s^4 + C_{y.3} \cdot s^3 + C_{y.2} \cdot s^2 + C_{y.1} \cdot s^1 + C_{y.0} \tag{4.29}$$

To determine the reference point, p_{ref} , the desired path is interpolated densely enough with the polynomial approximation along the curvilinear abscissa as follow:

$$P_{des} = \left\{ p_{des.i} \middle| i = 1, \cdots, N_{des} \right\} \tag{4.30}$$

$$p_{des.i} = \left[x_{des}(s_i) \quad y_{des}(s_i) \right]^T \in \mathbb{R}^{2\times 1} \tag{4.31}$$

$$S_{arr} = \left\{ s_i \middle| i = 1, \cdots, N_{des} \right\} \tag{4.32}$$

The reference point, P_{ref} can be determined technically as follow:

$$p_{ref} = \underset{p_{des.i} \in P_{des}}{\arg\min} \left\| p_{des.i} - p_{veh} \right\| \tag{4.33}$$

$$p_{veh} = \left[p_X \quad p_Y \right]^T \in \mathbb{R}^{2\times 1} \tag{4.34}$$

where P_{veh} is position vector of the CG point of the vehicle with p_X , p_Y coordinates.

Similarly, the reference point in the future, $P_{ref.p}$ can be determined from the endpoint, $P_{veh.p}$ of the predicted future driving trajectory.

$$p_{ref.p} = \underset{p_{des.i} \in P_{des}}{\arg\min} \left\| p_{des.i} - p_{veh.p} \right\| \tag{4.35}$$

$$p_{veh.p} = \left[p_{X.p} \quad p_{Y.p} \right]^T \in \mathbb{R}^{2\times 1} \tag{4.36}$$

If the reference point in the desired trajectory is observed in the vehicle

coordinate, it can be written as:

$$P_{ref.veh} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}^{-1} \cdot \left(P_{ref} - P_{veh} \right) \tag{4.37}$$

Let y-coordinate component of $P_{ref.veh}$ be $P_{ref.veh.y}$.

The current lateral error can be defined as follow:

$$e_y = -sgn\left(P_{ref.veh.y} \right) \cdot \left\| P_{veh} - P_{ref} \right\| \tag{4.38}$$

Similarly, the predicted lateral error can be defined as follow:

$$e_{y.p} = -sgn\left(P_{ref.p.veh.p.y} \right) \cdot \left\| P_{veh.p} - P_{ref.p} \right\| \tag{4.39}$$

where $P_{ref.p.veh.p.y}$ is the y-coordinate component of $P_{ref.p.veh.p}$ which stands for the reference point in the future observed from the predicted vehicle coordinate in the future.

The cosine and sine of the heading error can be calculated as follow:

$$\begin{bmatrix} \cos(e_\theta) \\ \sin(e_\theta) \end{bmatrix} = \begin{bmatrix} \mathbf{T}_{ref} & \mathbf{N}_{ref} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \cos(\psi) \\ \sin(\psi) \end{bmatrix} \tag{4.40}$$

$$\mathbf{T}_{ref} = \begin{bmatrix} \cos(\psi_d) \\ \sin(\psi_d) \end{bmatrix} \in \mathbb{R}^{2\times 1} \tag{4.41}$$

$$\mathbf{N}_{ref} = \begin{bmatrix} \cos\left(\frac{\pi}{2}\right) & -\sin\left(\frac{\pi}{2}\right) \\ \sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{bmatrix} \cdot \mathbf{T}_{ref} \in \mathbb{R}^{2\times 1} \tag{4.42}$$

where, $\mathbf{T}_{ref} \in \mathbb{R}^{2\times1}$ is the unit tangential vector at the reference point, P_{ref}, and $\mathbf{N}_{ref} \in \mathbb{R}^{2\times1}$ is the unit normal vector at the identical point.

Similarly, the cosine and sine of the predicted heading error can be calculated as follow:

$$\begin{bmatrix} \cos\left(e_{\theta.p}\right) \\ \sin\left(e_{\theta.p}\right) \end{bmatrix} = \begin{bmatrix} \mathbf{T}_{ref.p} & \mathbf{N}_{ref.p} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \cos\left(\psi_p\right) \\ \sin\left(\psi_p\right) \end{bmatrix} \tag{4.43}$$

where ψ_p is the predicted heading of the vehicle at the endpoint of the predicted future driving trajectory. $\mathbf{T}_{ref.p} \in \mathbb{R}^{2\times1}$ is the unit tangential vector at the reference point in the future, $P_{ref.p}$. And $\mathbf{N}_{ref.p} \in \mathbb{R}^{2\times1}$ is the unit normal vector at $P_{ref.p}$. The predicted heading error, $e_{\theta.p}$ can be defined as:

$$e_{\theta.p} = \psi_p - \psi_{d.p} \tag{4.44}$$

From the above, the tracking error including the current and the predicted error can be defined completely.

4.3. Desired Motion Decision for Path-Tracking Error Regulation

The path-tracking can be considered as the error regulation problem. For the path-tracking error regulation, the desired yaw-rate of the vehicle is determined by the synthetic input technique. The synthetic input technique is employed in numerous previous research (Sanketi et al., 2005), (Song et al., 2014), (Swaroop, Hedrick, Yip, & Gerdes, 2000). From the synthetic input technique, the controller design procedure can be divided into two parts: the desired motion decision and the desired motion tracking. The synthetic input is the desired motion of the vehicle to track the desired path, and the desired steering wheel angle as a control input is determined to track the desired motion from the assumed dynamics of the vehicle. In this layered architecture of a controller, various types of vehicle dynamics models can be considered to be applied. On the other hand, (Rajamani, 2011) derived the path-tracking error dynamics from coupling the vehicle dynamics and the path-error states. The vehicle dynamics model, which was applied to, is a planar single-track model with a linear tire model.

As one of the main contribution of this dissertation, the combined lateral error is defined using the predicted lateral error as follows:

$$e_{y.com}(t) = e_y(t) + \rho_p \cdot e_{y.p}(t) \tag{4.45}$$

where $e_{y.com}$ is the combined lateral error, and ρ_p is weighting factor of

the predicted lateral error.

The derivative and 2nd derivative of the combined lateral error can be written as follow:

$$\dot{e}_{y.com}(t) = v_x(t)\sin\left(e_\theta(t)\right) + \rho_p \cdot \dot{e}_{y.p}(t) \tag{4.46}$$

$$\begin{aligned}\ddot{e}_{y.com}(t) = \Big[& \dot{v}_x(t)\sin\left(e_\theta(t)\right) + \\ & v_x(t)\cos\left(e_\theta(t)\right)\cdot\left\{\gamma(t) - \kappa(t)\cdot v_x(t)\cos\left(e_\theta(t)\right)\right\}\Big]\end{aligned} \tag{4.47}$$

The 2nd derivative of the predicted lateral error, which is highly nonlinear term is assumed to be negligible.

The sliding surface is defined as follow:

$$S(t) = \dot{e}_{y.com}(t) + \rho \cdot e_{y.com}(t) \tag{4.48}$$

$$\dot{S}(t) = \ddot{e}_{y.com}(t) + \rho \cdot \dot{e}_{y.com}(t) \tag{4.49}$$

The yaw-rate of the vehicle is treated as a synthetic input in this dissertation. The synthetic input is designed to impose the stable first order dynamics on the sliding surface. Consequently, the sliding surface asymptotically converge to zero with the synthetic input. The synthetic input can be obtained as follow:

$$\begin{aligned}\gamma_{syn}(t) = \frac{1}{v_x(t)\cdot\cos(e_\theta(t))}\cdot\Big[& -\left(\lambda+\rho\right)\cdot\dot{e}_{y.com}(t) - \lambda\cdot\rho\cdot e_{y.com}(t) \\ & - \dot{v}_x(t)\sin(e_\theta(t))\Big] + \kappa(t)\cdot v_x(t)\cos(e_\theta(t))\end{aligned} \tag{4.50}$$

where γ_{syn} is the yaw-rate as the synthetic input. λ is positive decaying ratio of the first order dynamics that the synthetic input impose on the sliding surface.

With neglecting the discrepancy between the synthetic input yaw-rate and

the actual yaw-rate of the vehicle, the sliding surface will follow the first order dynamics as:

$$\dot{S}(t) = -\lambda \cdot S(t) \tag{4.51}$$

A Lyapunov function candidate can be defined as:

$$V(t) = \frac{1}{2} \cdot S^2(t) \tag{4.52}$$

Differentiating the Lyapunov function candidate Eq.(4.52),

$$\dot{V}(t) = S(t) \cdot \dot{S}(t) = -\lambda \cdot S^2(t) \leq 0 \tag{4.53}$$

An architecture of the desired motion decision for the desired path tracking is illustrated in Figure 4.4. The dotted-grey boundary box refers to the design parameters. The inputs of the module are the desired path, the pose information of the vehicle, and the dynamic states of the vehicle such as the longitudinal velocity, acceleration, steering wheel angle.

Note that the vehicle model parameter comes from online vehicle motion learning module. The pose information of the vehicle is obtained from the localization module.

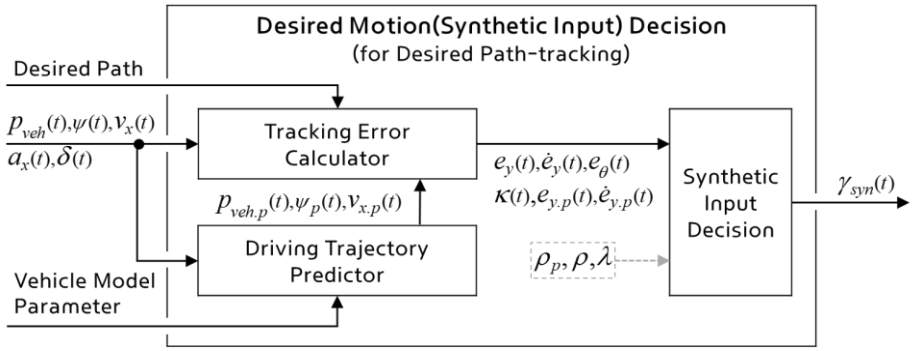

Figure 4.4. An architecture of the desired motion decision for the desired path tracking

4.4. Model Reference Adaptive Control for Desired Motion Tracking

As mentioned in chapter 4.3, the proposed steering controller has layered architecture. The dynamic surface control (DSC) method, which employs this architecture, has been proposed by (Sanketi et al., 2005; Song et al., 2014; Swaroop et al., 2000). A main contribution of the dynamic surface control method is an avoidance of "explosion of terms" problem due to the serial analytic derivative. The derivative of the synthetic input is needed in obtaining the control input to track the synthetic input. Instead of the analytic derivative, or forward Euler approximation, a derivative of the synthetic input is obtained from passing through a 1^{st}-order low pass filter in the dynamic surface control method.

However, the dynamic surface control method is able to be applied under the dynamic system with known parameters. It implies that the parameters in the vehicle model has to be identified with respect to a target vehicle model. If a single-track vehicle model with a linear tire model is employed, a number of parameters including cornering stiffness of tires, moment of inertia, location of the CG point, etc. have to be identified. However, some parameters are unmeasurable directly, and a number of principle test data is required for the identification.

As one of the main contribution of this dissertation, it has been intended to enhance the applicability of the proposed steering controller. In order to reduce

an amount of time that applying the controller to various vehicle model, the model reference adaptive control (MRAC) method is employed in this dissertation.

(Stepanyan & Krishnakumar, 2010) proposed modification of the conventional model reference adaptive control architecture. The MRAC method is able to be applied to an uncertain dynamics system. It has also been proved that the MRAC guaranteed tracking performance in both transient and steady-state behavior of the system.

In the MRAC, a reference model to be tracked can be defined as follow:

$$\begin{aligned} x_{ref}(0) &= x(0) \\ \dot{x}_{ref}(t) &= -a_{ref} \cdot x_{ref}(t) + b_{ref} \cdot r(t) + \lambda_{ref} \cdot \left\{ x(t) - x_{ref}(t) \right\} \end{aligned} \tag{4.54}$$

where a_{ref} , b_{ref} are positive constant of the reference model parameters. λ_{ref} is a design parameter. The parameters of a reference model are chosen to meet the specified. $x_{ref}(t)$ is state of the reference model dynamics, and $r(t)$ is assumed to be a smooth and bounded desired state. In this case $r(t)$ is the desired yaw-rate of the vehicle, which is determined in the chapter 4.3.

$$r(t) = \gamma_{syn}(t) \tag{4.55}$$

Note that the reference model in Eq.(4.54) can be considered as a 1st-order low pass filter for the desired state, $r(t)$ when the reference model trajectory is tracked perfectly by the system trajectory. That is similar with the dynamic

surface control method. Accordingly, the desired yaw-rate as the synthetic input can be modified to become more smooth.

Eq.(4.54) is proposed by (Stepanyan & Krishnakumar, 2010). In the design of the reference model, the discrepancy between the state of the dynamic system and the reference model is incorporated for the better tracking performance in transient. The discrepancy term in the reference model drives itself toward the system proportionally to the tracking error in transient.

The dynamical system to be controlled is defined as a first order linear system:

$$\dot{x}(t) = a \cdot x(t) + b \cdot u(t) \tag{4.56}$$

where $x(t) \in \mathbb{R}^{1\times 1}$ and $u(t) \in \mathbb{R}^{1\times 1}$ are the state and the input of the system respectively. Note that a and b are unknown constant parameters with known sign of b.

There have been several models of lateral vehicle dynamics that represent the relationship between the front steering angle and the yaw-rate of the vehicle. Besides methods which requires numerous model parameters, it is well known that an approximated 1st-order system with the yaw-rate gain is reasonable for the steering control in previous research (Abe, 2015; Jung, Kim, Son, Lee, & Yi, 2014; Wong, 2001).

$$\gamma(s) = \frac{G_\gamma}{s + \frac{1}{\tau}} \cdot \frac{1}{G_{str}} \cdot \delta(s) \tag{4.57}$$

where s is the Laplace transform operator, and $\gamma(s), \delta(s)$ are the Laplace transforms for the yaw-rate and the steering wheel angle respectively. G_γ is the

yaw-rate gain, and G_{str} is the steering ratio between the steering wheel angle and front steering angle.

Assuming the initial values of states are zero, Eq.(4.57) can be re-written in time domain as follow:

$$\underset{\dot{x}(t)}{\dot{\gamma}(t)} = \underbrace{-\frac{1}{\tau}}_{a} \cdot \underset{x(t)}{\gamma(t)} + \underbrace{\frac{G_{\gamma}}{G_{str}}}_{b} \cdot \underset{u(t)}{\delta(t)} \tag{4.58}$$

It can be seen that Eq.(4.58) is in the form of Eq.(4.56). Consequently, there is no need to identify the model parameters in Eq.(4.58), because the MRAC is applied to the uncertain system.

It is known that the behavior of the system in Eq.(4.56) tracks the modified desired state using the ideal control signal as follow:

$$u^*(t) = k_1^* \cdot x_{ref}(t) + k_2^* \cdot r(t) \tag{4.59}$$

where k_1^*, k_2^* are the ideal control gains, and have to satisfy the matching conditions as follow:

$$b \cdot k_1^* = -a - a_{ref} \tag{4.60}$$

$$bk_2^* = b_{ref} \tag{4.61}$$

The ideal control signal is unable to be implemented in the form of Eq.(4.59) because of the unknown model parameters, a and b. Accordingly, the modified control input with the estimates of the ideal control gains as follow:

$$u(t) = \hat{k}_1(t) \cdot x(t) + \hat{k}_2(t) \cdot r(t) \tag{4.62}$$

where $\hat{k}_1(t), \hat{k}_2(t)$ are the estimates of the ideal control gains. The estimates are updated by the adaptive laws as follows:

$$\dot{\hat{k}}_1(t) = -\gamma_{adpt} \cdot x(t) \cdot \left\{ x(t) - x_{ref}(t) \right\} \tag{4.63}$$

$$\dot{\hat{k}}_2(t) = -\gamma_{adpt} \cdot r(t) \cdot \left\{ x(t) - x_{ref}(t) \right\} \tag{4.64}$$

where $\gamma_{adpt} > 0$ is the adaptation rate.

In most cases, it is unavoidable that the high adaptation rate causes high frequency oscillations of the system. However, incorporating the discrepancy between the system and the reference model into the reference model design prevents the high frequency oscillations. It is proved that the closed-loop system with the above MRAC method is stable not only asymptotically but also in transient (Stepanyan & Krishnakumar, 2010).

In summary, the desired steering wheel angle to track the desired yaw-rate of the vehicle is obtained from Eq.(4.62).

$$\delta_{des}(t) = \hat{k}_1(t) \cdot \gamma(t) + \hat{k}_2(t) \cdot \gamma_{syn}(t) \tag{4.65}$$

The reference model and the adaptation law in Eq.(4.54), Eq.(4.63), and Eq.(4.64) can be re-written as follows:

$$\begin{aligned} &\gamma_{ref}(0) = \gamma(0) \\ &\dot{\gamma}_{ref}(t) = -a_{ref} \cdot \gamma_{ref}(t) + b_{ref} \cdot \gamma_{syn}(t) + \lambda_{ref} \cdot \left\{ \gamma(t) - \gamma_{ref}(t) \right\} \end{aligned} \tag{4.66}$$

$$\dot{\hat{k}}_1(t) = -\gamma_{adpt} \cdot \gamma(t) \cdot \left\{ \gamma(t) - \gamma_{ref}(t) \right\} \tag{4.67}$$

$$\dot{\hat{k}}_2(t) = -\gamma_{adpt} \cdot \gamma_{syn}(t) \cdot \left\{ \gamma(t) - \gamma_{ref}(t) \right\} \tag{4.68}$$

An architecture of the model reference adaptive controller for the desired yaw-rate tracking is illustrated in Figure 4.5. The dotted-grey boundary box refers to the design parameters. The inputs of the module are the desired and actual yaw-rate of the vehicle.

Note that the desired steering angle for the desired motion tracking is calculated without the prior-knowledge of the vehicle model parameter.

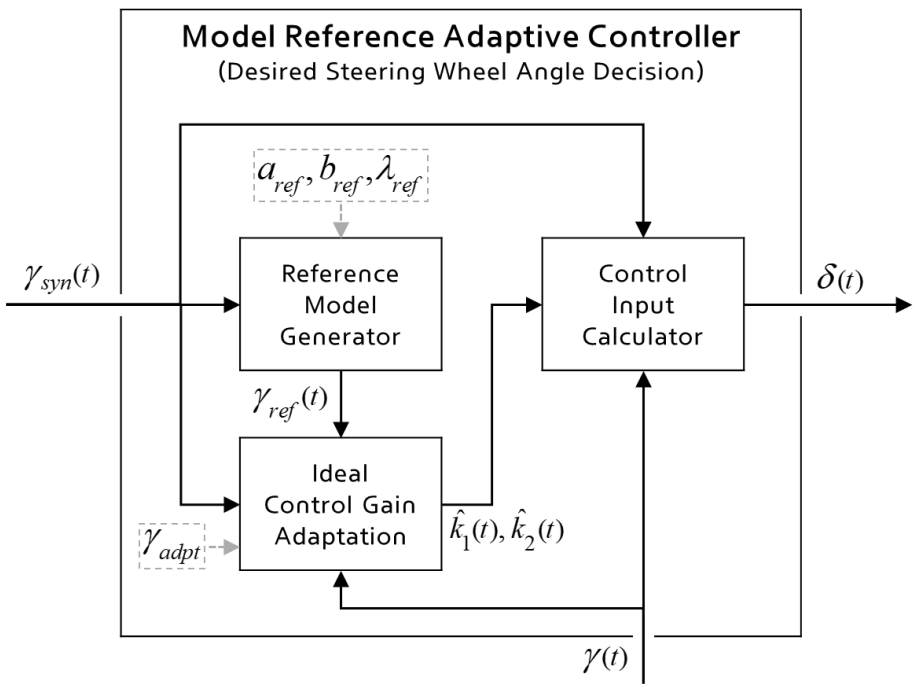

Figure 4.5. An architecture of the model reference adaptive controller for the desired yaw-rate tracking

Chapter 5

Simulation based Performance Evaluation

The proposed steering controller has been evaluated via simulation tests. The model predictive control (MPC) approach has been compared with the proposed steering controller in terms of computation load. Similar to MPC approach, the proposed steering controller utilizes the prediction information. However, the proposed steering controller determines the steering wheel angle with non-optimization approach. In contrast with the proposed steering controller, the MPC approach solves the optimization problem in every time step. The simulation results show that the proposed steering controller can achieve the desirable tracking performance with much lower computation load than the MPC approach.

One of the main contribution of the proposed steering controller is the online vehicle model learning feature. From the simulation test, how the vehicle model, which is identified online with incremental learning approach, is evolving is

presented. The online learning of the vehicle model is conducted under the standardized ISO/TR 8725 scenario.

Utilizing the predicted future error is one of the main contribution of the proposed steering controller. The effect of the predicted future error is validated via the U-turn scenario simulation.

5.1. Comparison with model predictive controller

The proposed steering controller has been compared with a model predictive control approach. The model predictive control approach has been applied to the lateral control in numerous researches. The model predictive control approach is based on the online optimization technique. In the optimization framework, the optimal state trajectories are generated as the optimal input profile in a certain preview time horizon. Therefore, the model predictive control approach has been utilized to cover the motion planning and control both. Notably, the lane change problem of an autonomous driving vehicle has been handled with the model predictive control approach widely.

To compare the desired trajectory tracking performance between the proposed steering controller with the model predictive control approach, a function of the model predictive controller is confined to the path-tracking in this research. The comparison has been conducted in view of computational efficiency mainly.

A single-track vehicle model with a linear tire model has been deployed for the model predictive control approach. The state of the model predictive

controller for the path-tracking has been defined as:

$$\boldsymbol{x}_{mpc} = \begin{bmatrix} v_y & \gamma & e_y & e_\theta \end{bmatrix}^T \in \mathbb{R}^{4\times 1} \tag{5.1}$$

The nonlinear differential equations, which describes the motion of the vehicle in the road-aligned coordinate frame can be written as:

$$\dot{v}_y = -\frac{2C_f + 2C_r}{Mv_x} \cdot v_y - \left(v_x + \frac{2C_f l_f - 2C_r l_r}{Mv_x} \right) \cdot \gamma + \frac{2C_f}{M} \cdot \delta_f \tag{5.2}$$

$$\dot{\gamma} = -\frac{2C_f l_f - 2C_r l_r}{I_z v_x} \cdot v_y - \frac{2C_f l_f^2 + 2C_r l_r^2}{I_z v_x} \cdot \gamma + \frac{2C_f l_f}{I_z} \cdot \delta_f \tag{5.3}$$

$$\dot{e}_y = v_x \cdot \sin\left(e_\theta\right) + v_y \cdot \cos\left(e_\theta\right) \tag{5.4}$$

$$\dot{e}_\theta = \gamma - \kappa \cdot v_x \tag{5.5}$$

It is assumed that the angular error e_θ is small enough, $\sin\left(e_\theta\right) \approx e_\theta$, $\cos\left(e_\theta\right) \approx 1$, and $\dot{s} \approx v_x$.

From the assumption, the nonlinear system dynamics (5.2) to (5.5) can be rearranged as:

$$\dot{\boldsymbol{x}}_{mpc}\left(t\right) = \mathbf{A}_{mpc}\left(\rho\left(t\right)\right) \cdot \boldsymbol{x}_{mpc}\left(t\right) + \mathbf{B}_{mpc} \cdot u_{mpc}\left(t\right) + \mathbf{B}_2 \cdot d_{mpc}\left(t\right) \tag{5.6}$$

where, $\rho\left(t\right) = \begin{bmatrix} v_x\left(t\right) & \kappa\left(t\right) \end{bmatrix}^T$ is the time-varying parameter, and $u_{mpc}\left(t\right) = \delta_f\left(t\right)$, $d_{mpc}\left(t\right) = \kappa \cdot v_x$.

$$\mathbf{A}_{mpc}=\begin{bmatrix} -\dfrac{2C_f+2C_r}{Mv_x} & -\left(v_x+\dfrac{2C_fl_f-2C_rl_r}{Mv_x}\right) & 0 & 0 \\ -\dfrac{2C_fl_f-2C_rl_r}{I_zv_x} & -\dfrac{2C_fl_f^2+2C_rl_r^2}{I_zv_x} & 0 & 0 \\ 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.7}$$

$$\mathbf{B}_{mpc}=\begin{bmatrix} \dfrac{2C_f}{M} & \dfrac{2C_fl_f}{I_z} & 0 & 0 \end{bmatrix}^T \tag{5.8}$$

$$\mathbf{B}_2=\begin{bmatrix} 0 & 0 & 0 & -1 \end{bmatrix}^T \tag{5.9}$$

For visual clarity, the system matrix $\mathbf{A}_{mpc}\left(\rho(t)\right)$ is expressed as $\mathbf{A}_{mpc}$.

The linear time-varying system, (5.6) is discretized by Euler's method as:

$$\boldsymbol{x}_{mpc.k}=\mathbf{A}_{d.mpc.k}\cdot\boldsymbol{x}_{mpc.k-1}+\mathbf{B}_{d.mpc}\cdot u_{mpc.k-1}+\mathbf{B}_{d.2}\cdot d_{mpc.k-1} \tag{5.10}$$

$$\mathbf{A}_{d.mpc.k}=\left(\mathbf{I}_{4\times4}+T_{s.mpc}\cdot\mathbf{A}_{mpc.k}\right) \tag{5.11}$$

$$\mathbf{B}_{d.mpc}=T_{s.mpc}\cdot\mathbf{B}_{mpc} \tag{5.12}$$

$$\mathbf{B}_{d.2}=T_{s.mpc}\cdot\mathbf{B}_2 \tag{5.13}$$

where, $T_{s.mpc}$ is a sample time, and is chosen as 0.1sec.

An output of the system is defined as:

$$\mathbf{y}_{mpc.k}=\begin{bmatrix} e_y & e_\theta \end{bmatrix}^T\in\mathbb{R}^{2\times1} \tag{5.14}$$

The optimal control sequence profile under the nonlinear system dynamics

(5.10) is obtained by solving a constrained quadratic optimization problem. The optimization problem which is solved repeatedly at each time step is formulated as follows:

$$\min_{u_{mpc.k}} \sum_{k=0}^{N_p-1} \left(\left\| \boldsymbol{y}_{mpc.k+1} \right\|_Q^2 + \left\| u_{mpc.k} \right\|_R^2 \right) \tag{5.15}$$

$$s.t. \quad \boldsymbol{x}_{mpc.k+1} = \mathbf{A}_{d.mpc.k} \cdot \boldsymbol{x}_{mpc.k} + \mathbf{B}_{d.mpc} \cdot u_{mpc.k} + \mathbf{B}_{d.2} \cdot d_{mpc.k} \tag{5.16}$$

$$\left\| u_{mpc.k} \right\| \le u_{\max} \tag{5.17}$$

$$\left\| u_{mpc.k+1} - u_{mpc.k} \right\| \le s_{u.mpc} \tag{5.18}$$

$$\boldsymbol{x}_{mpc.0} = \boldsymbol{x}_{mpc}(t) \tag{5.19}$$

$$k = 0, \ldots, N_p - 1$$

where, $\mathbf{Q}$ and $\mathbf{R}$ are the weighting matrices.

The vehicle model parameters used in (5.10), are listed in Table 5-1.

Table 5-1. The vehicle model parameters for the model predictive control design

Symbol	Value	Unit
C_f	55000	N/rad
C_r	60000	N/rad
M	1500	kg
l_f	1.215	m
l_r	1.485	m

I_z	2778	kgm^2

The above optimization problem is solved at each step and the input which is applied to the vehicle model at current step is determined as the first term of the optimal control sequences.

The nonlinear model predictive control solver, which is given in MATLAB/Simulink, has been utilized to solve the optimization problem.

As mentioned before, the comparison with the MPC approach has been conducted in view of computational load. The standardized closed-loop test scenario derived from ISO 3888-1:1999 has been utilized. The ISO 3888 specifies the dimensions of the test track for a double lane change scenario. The original objective of the double lane change scenario test is to identify road-holding ability of passenger cars. However the simulation was conducted under nominal driving condition ($a_y \leq 0.3g$). The dimension of the test track is illustrated in Figure 5.1. In actual test, a driving trajectory is not specified in the ISO 3888, and it is left to driver's discretion. However, for the trajectory tracking simulation, an arbitrary trajectory was planned prior to the simulation. The desired trajectory is illustrated in a dotted black line in Figure 5.1, and the curvature profile of the desired trajectory is also depicted in Figure 5.1. The minimum turning radius of the desired trajectory is $50m$. That is similar level of the section 4 in the actual vehicle test course.

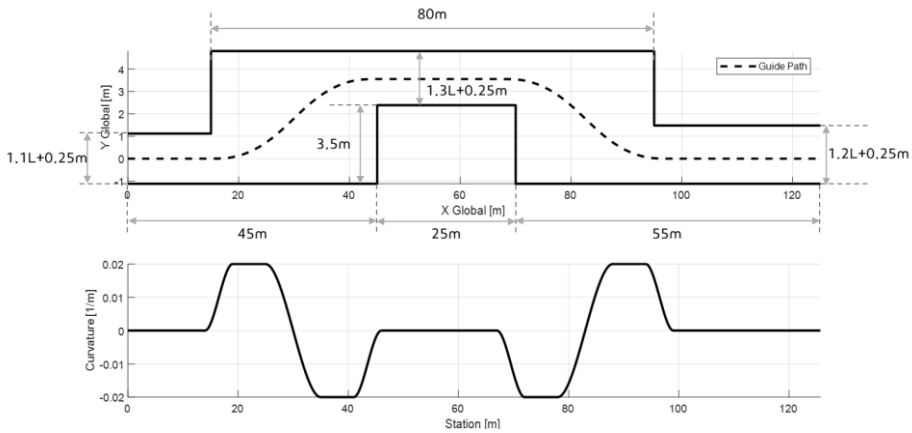

Figure 5.1. Test track dimension, specified in ISO 3888-1:1999

The comparison simulation results are presented in Figure 5.2. A solid red line represents the states of a vehicle model, which is controlled by the proposed steering controller. The tracking performance results of the MPC are illustrated in a dash-dotted blue line. The vehicle speed is set to be 40km/h, and the maximum lateral acceleration is under 0.3g. The MPC preview horizon is configured to be 1sec, and the sampling time is 0.1sec. The tracking performance of the proposed steering controller is better than that of the MPC. However, both controllers manipulated the vehicle model to stay within 0.3m from the desired trajectory.

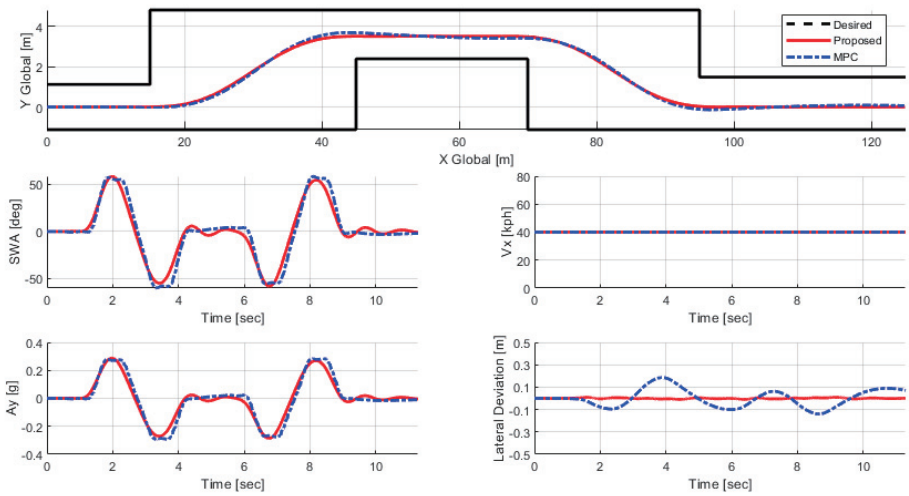

Figure 5.2. The comparison simulation results in double lane change scenario: The proposed steering controller vs the MPC approach

As mentioned above, this comparison is conducted in view of computational load. The simulation is conducted with a desktop PC, and the desktop PC has a processor of Intel(R) Core(TM) i7-9700K CPU @3.60GHz with 32.0GB RAM. The proposed steering controller and MPC-based controller have been simulated simultaneously. The simulation was conducted with MATLAB/Simulink.

The average computation time at each function module call is obtained by the simulation profile report provided in MATLAB/Simulink. The average computation time per function call has been recorded for 30 times of simulations. The test results are presented in Figure 5.3.

It was found that the MPC-based controller takes 93.3ms per function call on average, and the proposed controller takes only 7.5ms per function call. From the results, it can be considered that the proposed steering controller is more effective approach than the MPC-based controller in view of the computational load. Even the tracking performance of the proposed steering controller is superior to the MPC-based controller.

The MPC-based controller can integrate the motion planning and control part because it optimizes the states of the system dynamics as well as the input sequences. Accordingly, it can be considered that utilizing the MPC-based controller only for the desired trajectory tracking problem is inefficient. In that sense, it is validated that the proposed approach is quite practical way for steering control in the architecture that has the motion planning and control part separately.

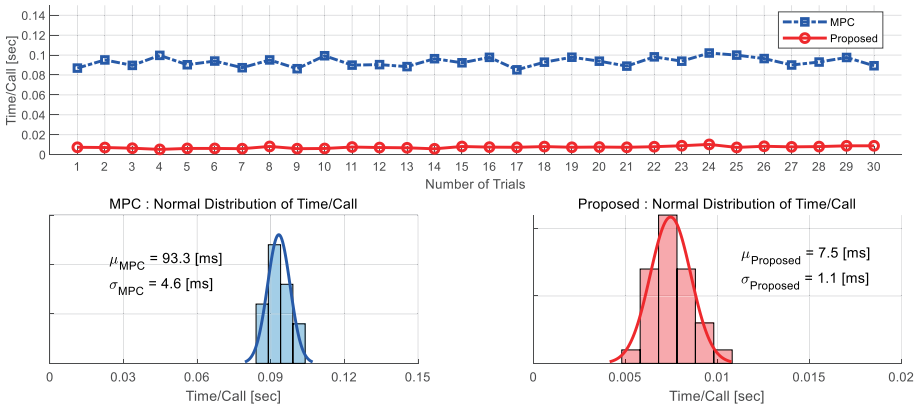

Figure 5.3. The comparison of the computation time per function module call between the proposed controller and MPC-based controller

5.2. Effect of incorporating prediction error

The proposed steering controller defines the tracking error, including the predicted error. That is one of the main contributions of this research. In order to identify the effect of the predicted error synthesis, the comparison test has been conducted with the proposed steering controller without the predicted error synthesis.

The predicted error is calculated from the predicted future trajectory of a vehicle. The predicted future trajectory functions similar to a parking guideline in a commercialized driver assistant system. When a driver reverses a vehicle equipped with a parking guide system, a guideline is shown in a cluster of a vehicle. The guideline bends with the steering wheel angle and assists the driver to manipulate the steering wheel. Similar to the parking guideline, even if the current vehicle position is identical, the error which occurs in a few milliseconds can vary with current steering behavior.

The predicted error term acts as a penalty term to prevent excessive steering or replenish deficient steering.

The simulation test scenario is U-turn in urban road. The turning radius of U-turn is set to be $5.3m$. The proposed steering controller without the predicted future error is compared with the original one. The driving trajectories are presented in Figure 5.4.

The result profiles of the proposed steering controller with the predicted

future error are illustrated as a solid red line. Section 1 in Figure 5.4 represents an entry of the U-turn. As shown in the steering wheel angle profile, the case without the predicted future error starts to rotate the steering wheel later than the case with the predicted future error. In the minimum turning radius scenario, such as the U-turn, the late-turning causes the late-releasing, as shown in section 2, because the maximum steering wheel angle has already been applied to the vehicle. In that case, the steering wheel is unable to start to release until the vehicle recovers proper position and orientation. In section 3, the case without the predicted future error shows the slight overshoot in the steering wheel angle. That steering wheel behavior causes the fluctuation of the lateral acceleration. The fluctuation of the lateral acceleration may cause uncomfortable ride quality for passengers.

The difference between the two cases results from the predicted future error compensation. A human driver controls the steering wheel from the current and the predicted states both. The proposed steering controller mimics the characteristics of the driving pattern of the human driver.

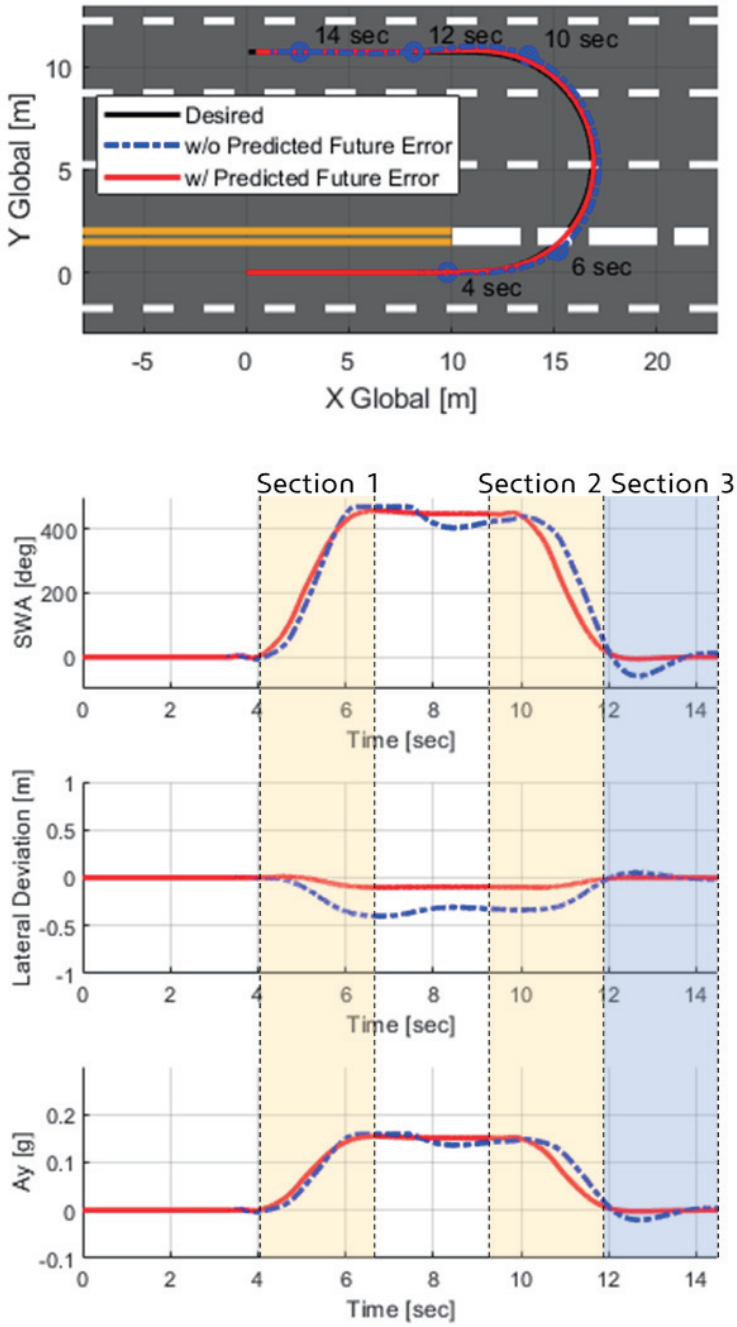

Figure 5.4. Effect of incorporating the predicted future error in the U-turn

5.3. Online vehicle model learning performance quantification

The predicted future trajectory is obtained from a vehicle model, which is identified online with the streaming driving data. The online vehicle model learning approach is one of the main contributions of this research.

The simulation test derived from the ISO/TR 8725:1988 has been conducted to quantify the online vehicle model learning performance. Most of the data-driven approach depends on what the learning data is utilized. Accordingly, there are no absolute criteria for the learning data.

The vehicle model learning module identifies the yaw-rate model of the vehicle from the steering wheel angle input and the velocity. In order to obtain the data pair from the standardized test scenario, the test method derived from the ISO/TR 8725:1988 has been applied to the simulation test. The ISO/TR 8725:1988 includes several transient open-loop response tests for the identification of transient behavior of a vehicle.

Among the test methods, one period sinusoidal steering input test has been applied. A single-track vehicle model with a linear tire model was utilized in the online learning simulation test. The conceptual diagram which describes the test method is presented in Figure 5.5. The detail test method is described in Table 5-2. The magnitude of the steering wheel angle which generates the target lateral acceleration in Table 5-2 has to be found.

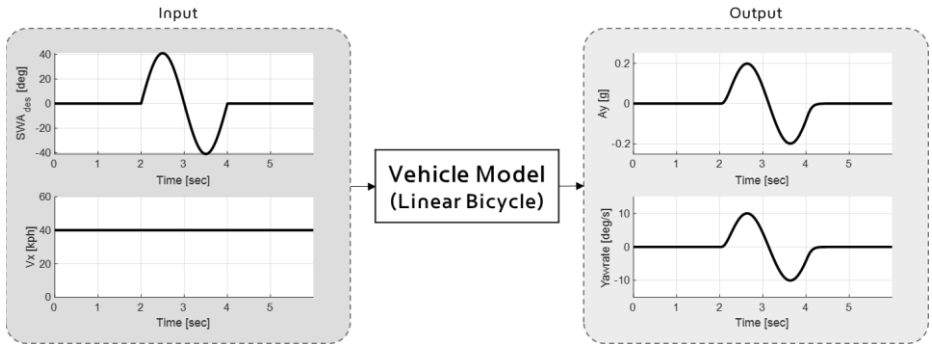

Figure 5.5. Conceptual diagram of one-period sinusoidal steering input test

Table 5-2. One-period sinusoidal steering input test of ISO/TR 8725:1988

	Standard	Optional
Magnitude of SWA	$a_y = 4m/s^2$	$a_y = 2m/s^2$, $a_y = 6m/s^2$
Frequency of SWA	0.5 Hz	1 Hz, 0.7 Hz, 0.3 Hz
Speed	100 km/h	Other test speeds of interest (20km/h steps)

From the specification of one-period sinusoidal steering input test in ISO/TR 8725:1988, there can be many scenario cases which are the combination of each condition, such as test speed and the magnitude/frequency of steering wheel angle. Accordingly, the simulation has been conducted with randomly shuffled scenario cases. 3 randomly shuffled scenario is considered as 1 epoch. 1 epoch takes about 10sec on average. It can vary with selected 3 frequency of the steering input profile. Online vehicle model learning simulation has been conducted with 200 epochs. 200 epochs may take about 30 minutes. However, the test speed in the original specification is over-speed in normal urban driving condition. Accordingly, the modified test method has been applied to the simulation. The modified test specification is described in Table 5-3.

Table 5-3. Modified specification of one-period sinusoidal steering input test

	Modified Specification
Magnitude of SWA	40°,50°,60°,70°,80°
Frequency of SWA	0.3Hz, 0.5Hz, 0.7Hz, 1Hz
Speed	20km/h, 40km/h, 60km/h

As mentioned before, a vehicle model is identified online by Incremental Sparse Spectrum Gaussian Process Regression (ISSGPR) method. How the Gaussian Process (GP) model is evolving through online learning process is presented in Figure 5.6 and Figure 5.7.

The closed-loop one-step ahead yaw-rate has been estimated by the online learned vehicle model. The estimated yaw-rate and the actual yaw-rate of a vehicle model in the simulation are presented as a dash-dotted red line and a solid black line. In order to depict the uncertainty of the estimated yaw-rate, the covariance boundary is plotted as a dashed blue line.

Figure 5.6 and Figure 5.7 are the snap-shots of the 25th and the 191st epoch, respectively. There can be found three-period steering input profile and desired speed profile in Figure 5.6 and Figure 5.7 because each epoch is composed of three randomly shuffled scenario.

As the performance index, which evaluates how the GP model evolved, the inner area of the covariance boundary of each epoch has been utilized. The more experience is stacked, the narrower the covariance boundary becomes. Therefore, it can be found that the covariance boundary area is decreased over the number of an epoch is increased in the comparison between the 25th and the 191st epoch.

Notably, the root-mean-square(RMS) yaw-rate error between the estimated and the actual is converged rapidly within a few numbers of an epoch. That is because the ISSGPR method is the data-driven method. Apart from how the GP model trusts the expected value, the expected value is located among the data the GP model experienced. That characteristic is important in view of

practicality. As mentioned before, the online learned vehicle model has to be working online. Locating the expected value among the previous experience at first and incrementally updating the degree of reliability can be a good strategy.

From the results, it has been validated that the proposed online learning approach identifies the yaw-rate model of a vehicle model online. The estimated yaw-rate represented the actual yaw-rate of a vehicle model. It is noted that there is no need the prior knowledge of the target vehicle dynamics.

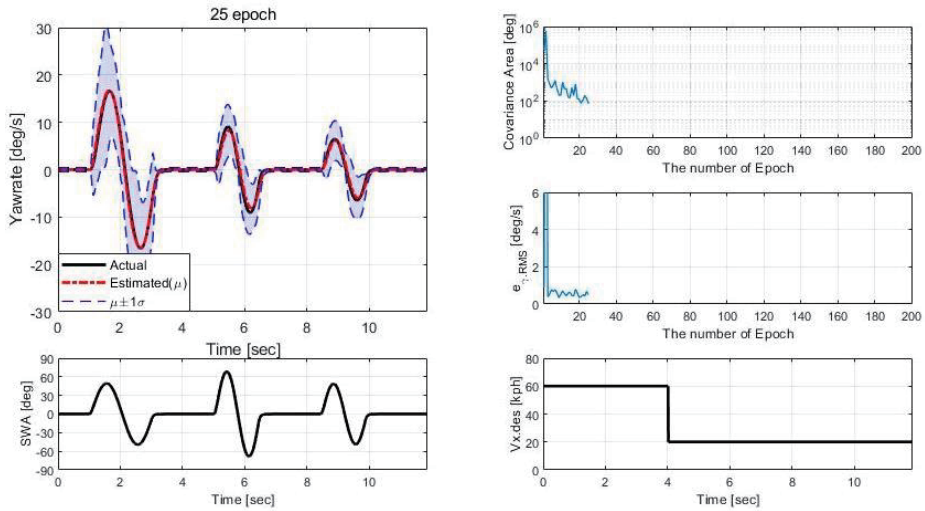

Figure 5.6. Closed-loop one-step ahead yaw-rate estimation results at the 25th epoch

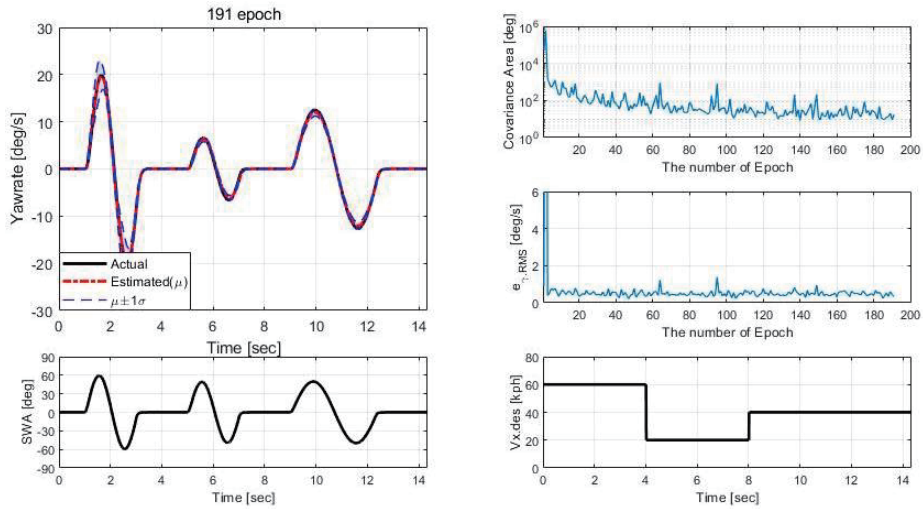

Figure 5.7. Closed-loop one-step ahead yaw-rate estimation results at the 191st epoch

Chapter 6

Vehicle Tests based Performance Evaluation

The proposed steering controller has been evaluated by vehicle tests. A B-segment SUV model (KIA NIRO hybrid) was utilized as the test vehicle. The desired trajectory tracking vehicle tests have been conducted at the internal circular road in Seoul National University, Korea. The given test course requires the steering controller to work properly in full-range of steering wheel angle, because the test course includes a minimum turning radius U-turn as well as a straight path.

It has been shown that the proposed steering controller tracks the given desired trajectory successfully. The test vehicle with the proposed steering controller drives the test course smoothly as like a human driver. The performance of the proposed controller has been compared with that of the pure-pursuit steering controller. The proposed controller shows not only better performance to track the desired trajectory than the pure-pursuit controller, but

also better applicability to various driving scenario without an additional parameter tuning.

The online vehicle model identification performance has also been evaluated in parallel. In order to verify how well the vehicle model represents the actual vehicle behavior, a closed-loop one-step ahead yaw-rate of the vehicle has been predicted in every time step.

It has been shown that the vehicle model, which is updated incrementally by the online learning approach, represents the behavior of the actual vehicle successfully. The online learning vehicle model is constructed by the streaming driving data only.

In order to verify the applicability of the online learning approach, a commercial van (Hyundai SOLATI) has also been utilized additional to the B-segment SUV (KIA NIRO). The verification with the two vehicles is quite reasonable, because the two vehicles are huge different in the size, and the dynamic characteristics. There was no need to adjust the learning parameter in application to another vehicle.

6.1. Configuration of Vehicle Tests

In order to conduct the vehicle test, the test vehicle has been built with some equipment. The test vehicle model is a B-segment SUV model(NIRO-hybrid) of KIA motors, Inc. The test vehicle is equipped with a Motor Driven Power Steering (MDPS), an electronic vacuum booster for brake-by-wire, and a throttle-by-wire. In order to build a vehicle control interface, a DriveKit, which is manufactured by PolySync, Inc., is installed in the test vehicle. The steering and the throttle/brake system can be manipulated by transmitting the control command values to the DriveKit.

An GPS-Aided Inertial Navigation System (INS) device, which is manufactured by Inertial Labs, Inc., is employed to measure global vehicle position information.

As a main computing controller, Micro-AutoBox, which is manufactured by dSpace Inc., is utilized with an industrial computer. The status of INS and Micro-AutoBox is monitored via the industrial computer.

The configuration of the test vehicle is presented in Figure 6.1

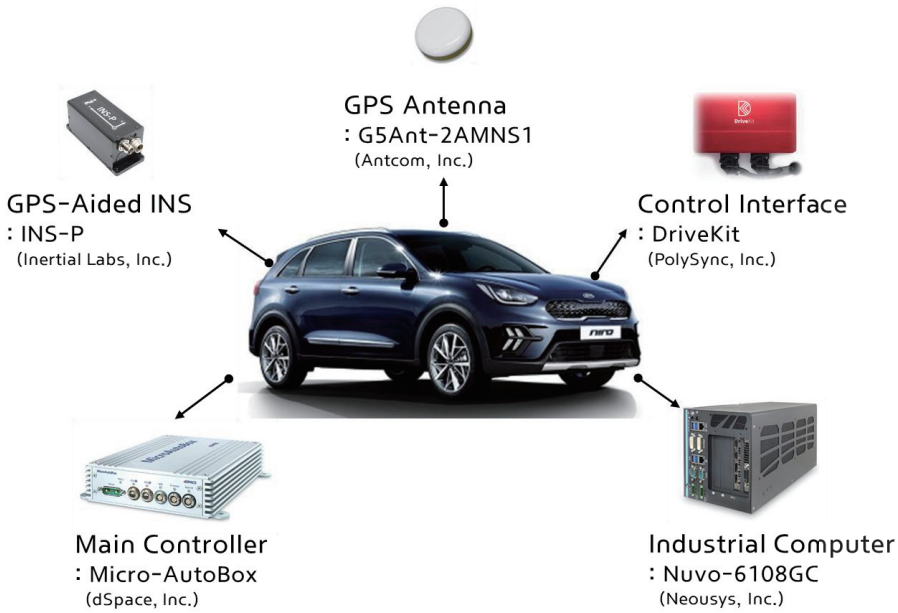

Figure 6.1. Test vehicle and Experimental setup

Figure 6.2 illustrates a connection diagram and data flow chart. As a main controller, the Micro-AutoBox collects the data from the vehicle and the INS-P through CAN communication, and transmits vehicle control command to the DriveKit. The Miro-AutoBox operates at a 100 Hz sampling rate.

The industrial computer receives Radio Technical Commission for Maritime Services(RTCM) data via Networked Transport of RTCM via Internet Protocol(NTRIP) client. NTRIP is a protocol for broadcasting the correction data. The industrial computer transmits DGPS correction information to INS-P to improve quality of positioning accuracy.

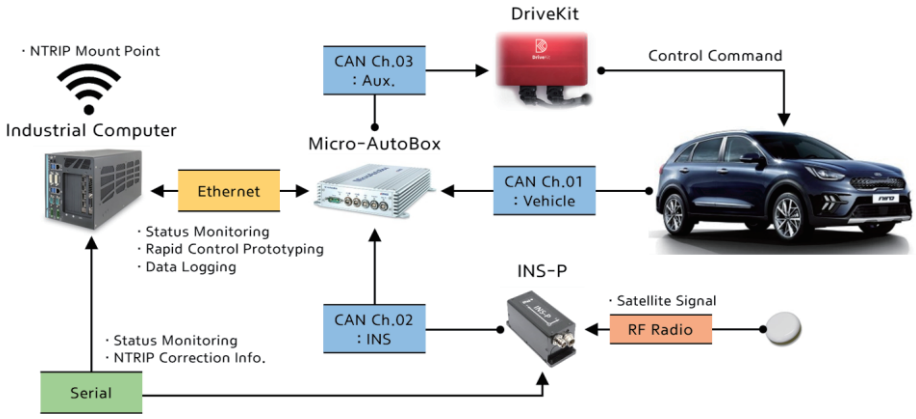

Figure 6.2. Connection diagram and data flow chart of the test vehicle

6.2. Vehicle Tests: Online Vehicle Motion Learning

Online vehicle motion learning performance is validated via vehicle test. Two types of vehicle are employed as target vehicle. One of the main objective of the proposed vehicle motion learning is to enhance applicability of the algorithm.

One-step ahead prediction results are compared between the proposed identification method and the conventional parametric modeling approach.

Note that the one-step ahead prediction results are obtained from the way, illustrated in Figure 6.3. The predicted yaw-rate is fed back to the input. Accordingly, the prediction quality affects the overall performance recursively.

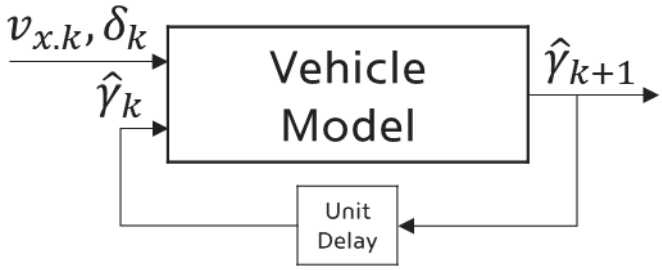

Figure 6.3. One-step ahead closed-loop prediction

Utilizing the actual yaw-rate in the current time to predict one-step ahead yaw-rate has to be excluded. Because, the identified vehicle model is unable to access the actual yaw-rate of the vehicle in prediction time horizon. The one-step ahead prediction with the actual yaw-rate in the current time step is able to be achieved from only Euler forward integration. In that case, there is no need to have identified vehicle model.

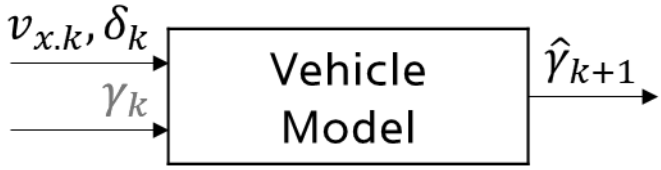

Figure 6.4. One-step ahead open-loop prediction

6.2.1. Target Vehicle Model: B-Segment SUV

As a target vehicle model, which is to be identified, B-segment SUV, KIA NIRO-hybrid has been utilized. The specification of the test vehicle is shown in Table 6-1. A moment of inertia of the vehicle is unrevealed parameter from a manufacturer. The moment of inertia is approximated from the assumption that the vehicle is a rectangular parallelepiped.

Table 6-1. Specification of Test Vehicle: KIA NIRO-hybrid

	Parameter	Unit	Value
Open	Mass	m	1550
	Total Length	m	4.355
	Wheelbase	m	2.7
	Total Width	m	1.805
Guess	Moment of Inertia	$kg \cdot m^2$	2871

The proposed identification method has been compared with a simple $\delta - \gamma$

model. In Ackermann steer model, the relationship between the steering wheel angle and the yaw-rate is known as Eq.(1.1). With the small angle and steady-state cornering assumption, Eq.(1.1) can be re-written as follow:

$$\delta_f = \frac{l_{wheelbase}}{R} = \frac{\gamma \cdot l_{wheelbase}}{v_x} \tag{6.1}$$

If the steering ratio and longitudinal velocity are assumed to be a constant, Eq.(6.1) can be re-written as follow:

$$\delta = \underbrace{\frac{G_{str} \cdot l_{wheelbase}}{v_x}}_{\text{constant}} \cdot \gamma \tag{6.2}$$

The simple gain model is based on Eq.(6.2). Even though the longitudinal velocity varies from zero to 60km/h, it can be considered that the simple gain assumption is reasonable within 100 degrees of the steering wheel angle, as shown in Figure 6.6.

The proportional gain between the steering wheel angle and the yaw-rate is estimated by weighted least square method. The blue solid line in Figure 6.6 is the simple gain model with an estimated gain.

Figure 6.5 denotes the yaw-rate response of the vehicle models. The dashed black line represents an actual yaw-rate response of the vehicle. The dashed-dotted red line denotes the vehicle model with the proposed method. The solid blue line represents the simple gain model. The vehicle test is carried out using full range of the steering wheel angle. As shown in Figure 6.5, the proposed identification method realizes the actual vehicle model perfectly. In contrast with the proposed vehicle model, the simple gain shows the trade-off characteristics in the yaw-rate response. The proportional gain seems to be

required to be larger within 10 deg/s yaw-rate. However, the gain seems to be excessive in more than 10 deg/s yaw-rate. It means that the simple gain model is only valid within piecewise range.

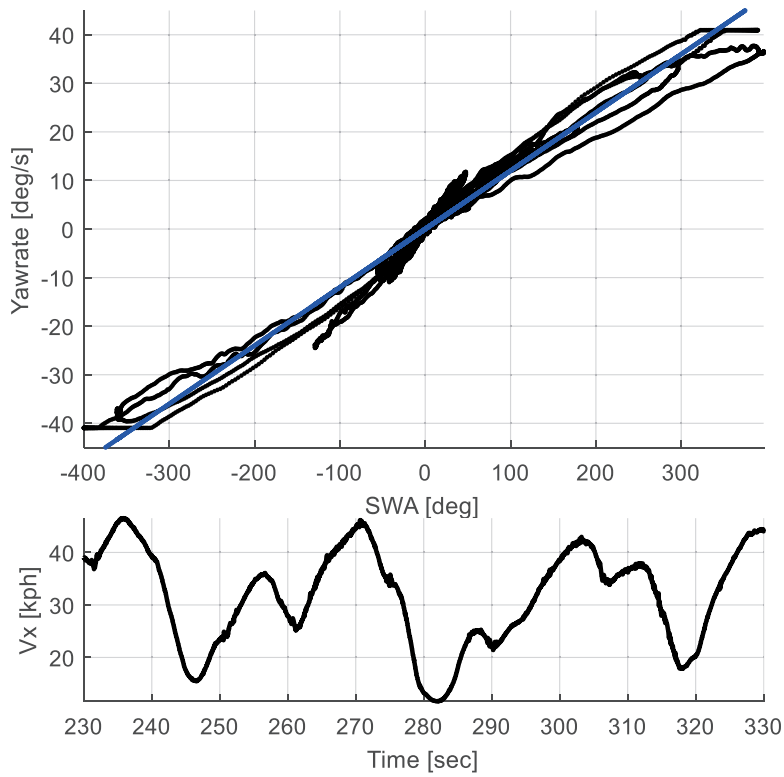

Figure 6.5. Identified vehicle model comparison: Simple gain to the steering wheel angle model

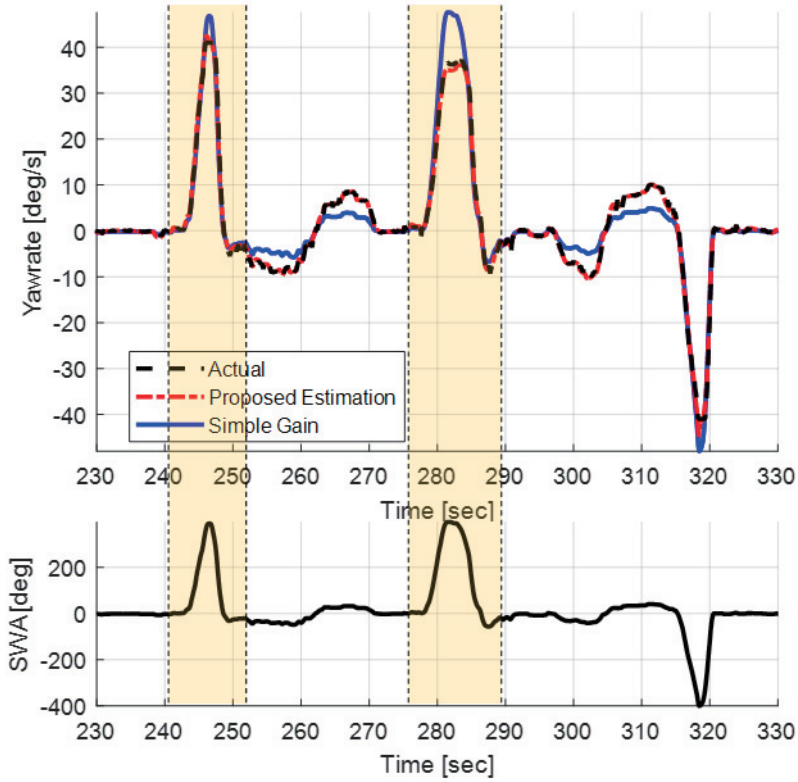

Figure 6.6. Identified vehicle model comparison: Simple gain from weighted least square method between the yaw-rate and the steering wheel angle

6.2.2. Target Vehicle Model: Commercial Van

The main objective of the proposed identification method is to enhance applicability of algorithm to various types of vehicle. Accordingly, the objective is validated via different types of vehicle case. The target vehicle is a commercial van, Hyundai SOLATI. The specification of SOLATI is listed in Table 6-2. As shown in Table 6-2, SOLATI is big different with NIRO in overall specification.

The vehicle test is conducted in autonomous shuttle route, which is designated by government. The shuttle route is shared with ordinary local buses and the site is located in Sangam-dong, Mapo-gu, Seoul, Korea.

Figure 6.7 shows the autonomous shuttle route in a map and GPS waypoints of the route, which is measured from driving with SOLATI.

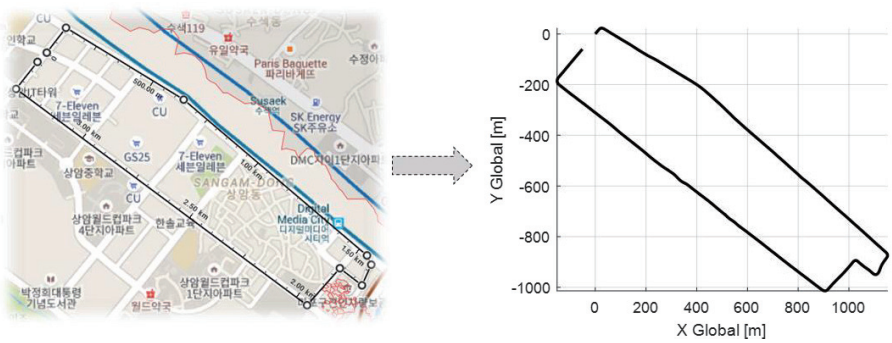

Figure 6.7. Driving course in urban scenario with a commercial van

Table 6-2. Specification of Test Vehicle: Hyundai SOLATI

	Parameter	Unit	Value
Open	Mass	m	4000
	Total length	m	6.195
	Wheelbase	m	3.67
	Total width	m	2.038
Guess	Moment of inertia	$kg \cdot m^2$	14886
	CG to front axle	m	1.468
	Steering ratio	-	12
	Front cornering stiffness	N/rad	25000
	Rear cornering stiffness	N/rad	30000

Figure 6.8 represents the vehicle states, measured from test driving in the autonomous shuttle route. As shown in the steering wheel angle profile, 6 right-angle turns were conducted. The longitudinal velocity varied from zero to 50km/h. The maximum yaw-rate of the vehicle was up to 40deg/s.

Note that the test driving covers the lane-change scenario and the right-angle turn. Thus, the steering wheel angle occurred up to 400deg.

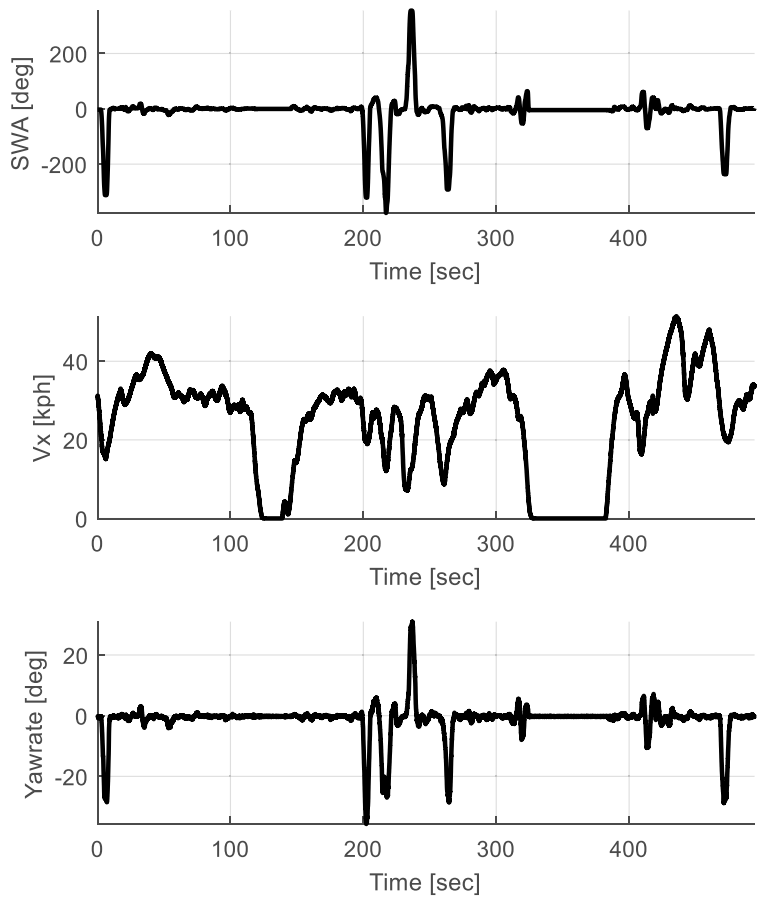

Figure 6.8. Driving data in urban with a commercial van: Steering wheel angle, longitudinal velocity, and yaw-rate

In this case, the proposed vehicle model identification method is compared with conventional parametric approach. The bicycle model with a linear tire model is widely used for vehicle control. The state-space model of the model is as follow:

$$\frac{d}{dt}\begin{bmatrix} v_y(t) \\ \gamma(t) \end{bmatrix} = \begin{bmatrix} -\dfrac{2C_f + 2C_r}{M_v \cdot v_x} & -v_x - \dfrac{2C_f l_f - 2C_r l_r}{M_v \cdot v_x} \\ -\dfrac{2C_f l_f - 2C_r l_r}{I_z \cdot v_x} & -\dfrac{2C_f l_f^2 + 2C_r l_r^2}{I_z \cdot v_x} \end{bmatrix} \begin{bmatrix} v_y(t) \\ \gamma(t) \end{bmatrix} + \begin{bmatrix} \dfrac{2C_f}{M_v} \\ \dfrac{2C_f l_f}{I_z} \end{bmatrix} \delta(t) \tag{6.3}$$

where, C_f, C_r are the cornering stiffness at front and rear tire respectively, and l_f, l_r are distance from CG point to the front and rear axle respectively. I_z is moment of inertia of the vehicle, and M_v is mass.

The model parameters are listed in Table 6-2. As shown in the table, the guess parameters, such as the cornering stiffness, are unable to be measured readily. Thus, vehicle parameter set, which realizes the well-matched vehicle model has to be found by repeated simulations.

Figure 6.9 and Figure 6.11 represent the identified vehicle model comparison at right-angle turn case. Figure 6.10 and Figure 6.12 depict the comparison at the lane-change case. The solid black line denotes an actual yaw-rate response of the vehicle. The dashed-dotted red line depicts the vehicle model response using the proposed method, ISSGPR. The dashed-dotted blue line represents the bicycle model from Eq.(6.3).

There are two parameter set for the bicycle model in the test driving case, because there is a trade-off in the yaw-rate response. If the bicycle model is identified with respect to the mild yaw-rate case (lane-change), yaw-rate response at right-angle turn is predicted quite higher than the actual vehicle

response. The other way, the bicycle model is identified with respect to the high yaw-rate case (right-angle turn), yaw-rate response of model is predicted lower than the actual vehicle.

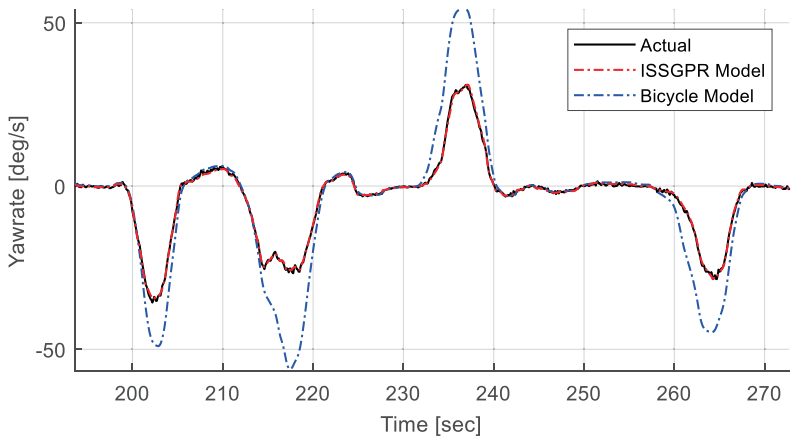

Figure 6.9. Identified vehicle model comparison at right-angle turn: Bicycle model with linear tire model for mild yaw-rate case

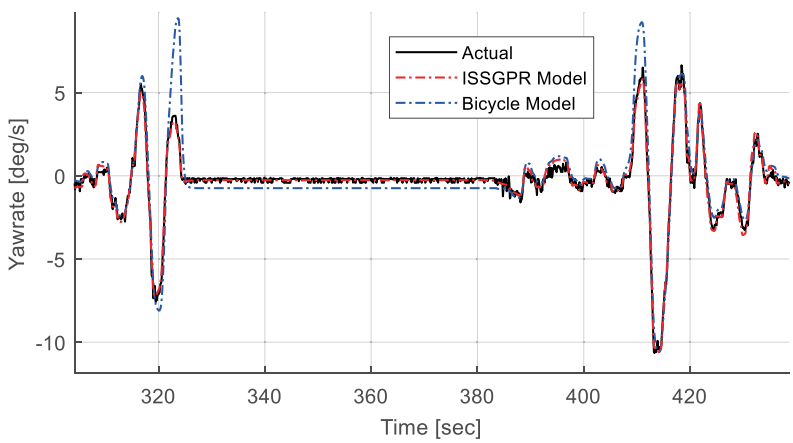

Figure 6.10. Identified vehicle model comparison at lane-change: Bicycle model with linear tire model for mild yaw-rate case

The trade-off characteristics can be described in mathematical view. The

state-space model in Eq.(6.3), is 2nd-order linear dynamic system. Accordingly, it can be re-formulated in terms of natural frequency and damping ratio. The natural frequency and the damping ratio are factors, which determines the system behavior. It is widely known that the natural frequency and the damping ratio are function of the cornering stiffness. It is shown that it can differ in phase delay characteristics at similar amplitude of the yaw-rate in Figure 6.11.

As seen in the simple gain model case, the yaw-rate response of the vehicle has non-linearity. Thus, it varies with the range of the yaw-rate. It can be also applicable to the bicycle model. Therefore, it is quite difficult to find well-matched model in general case.

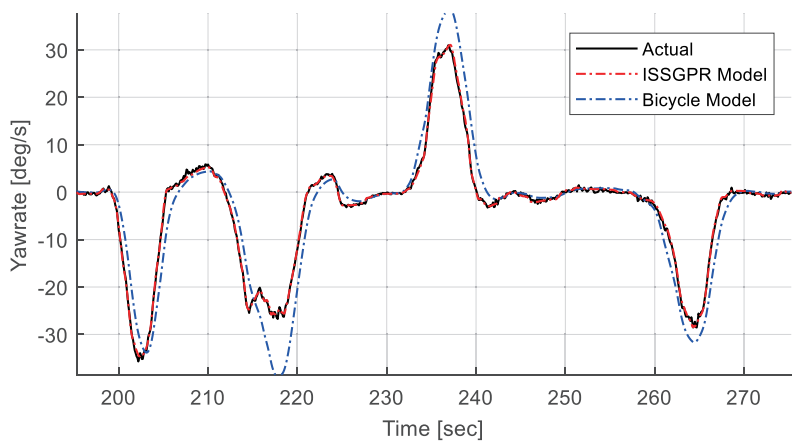

Figure 6.11. Identified vehicle model comparison at right-angle turn: Bicycle model with linear tire model for high yaw-rate case

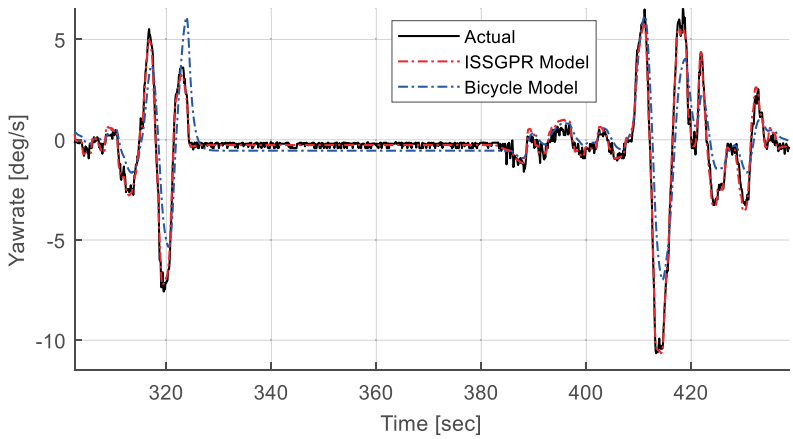

Figure 6.12. Identified vehicle model comparison at lane-change: Bicycle model with linear tire model for high yaw-rate case

6.3. Vehicle Tests: Trajectory Tracking on Urban Roads

The online vehicle motion learning-based steering controller, which is proposed in this dissertation, has been validated via a path-tracking vehicle test. The proposed steering controller is compared with the pure-pursuit steering controller, which is widely utilized in an automated driving steering control.

The vehicle tests have been conducted several times at the internal road at Seoul National University. The given test roads have quite complicated environments to drive automatically. The test course requires the steering controller to exploit the steering wheel angle up to the maximum for U-turn.

The global route of the test course is presented in Figure 6.13. The total distance of the test course is 1.23km, and the detail description of the test course is summarized in Table 6-3. As shown in Table 6-3, the curvature profile varies up to the minimum turning radius of the vehicle. Accordingly, a steering controller has to manipulate the steering wheel up to the maximum. The steering controller also has to secure straight-driving performance as well.

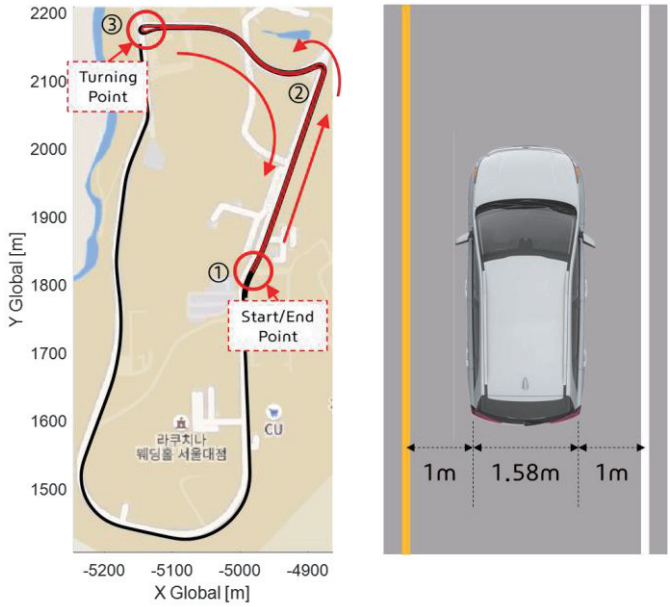

Figure 6.13. The vehicle test course and the road width in Seoul National University

Table 6-3. The detail description of the vehicle test course

Section No.	Route	Description	Remark
1	①	Start/End point	-
2	①→②	Straight down hill	Distance : 320m
3	②	Severe left turn	Radius : 6~7m
4	②→③	Winding course	Radius : 50m
5	③	U-Turn	Radius : 4~5m
6	③→②	Winding course	Radius : 50m
7	②	Severe right turn	Radius : 6~7m
8	②→①	Straight up hill	Distance : 320m

The performance of the steering control varies with driving speed. To identify the performance of the steering controller, objectively, the desired trajectory tracking test has been conducted with a desired speed tracking controller. The speed controller utilized in this research is proposed by (Kim, Kim, Shu, & Yi, 2015). The desired speed and the actual longitudinal velocity are depicted together in the 4th graph from the top in Figure 6.14.

The desired trajectory tracking test result is presented in Figure 6.14. The allowable lateral error in the test road is within $\pm 1m$. Lane departure occurs when the magnitude of the lateral error exceeds $1m$. The lateral error is calculated from the center of gravity point of the vehicle. The proposed steering controller kept the lateral error less than the allowable lateral error in the entire course.

Notably, the proposed steering controller made the test vehicle to stay within $0.4m$ from the desired trajectory in the straight driving scenario and U-turn scenario both.

As mentioned before, the desired trajectory was planned to require the maximum steering wheel angle, and it can be seen that the steering wheel rotated up to the maximum steering angle, 480° at the U-turn scenario. The winding course also requires the steering wheel angle up to 300°. There is no fluctuation in the steering wheel angle profile. That means the steering wheel rotated smoothly, and excessive steering was not applied. Excessive steering causes the unintended lateral error, and it leads to the counter correction steering. That behavior of the steering wheel makes the fluctuation of the steering wheel angle profile.

The desired yaw-rate tracking performance is also depicted in the 3rd graph from the top in Figure 6.14. The desired yaw-rate and the actual yaw-rate are presented in a red dash-dotted line and a solid black line, respectively. The desired yaw-rate for the desired trajectory tracking is tracked well by the model reference adaptive controller, described in 4.4. The desired yaw-rate tracking performance influences the desired trajectory tracking performance directly.

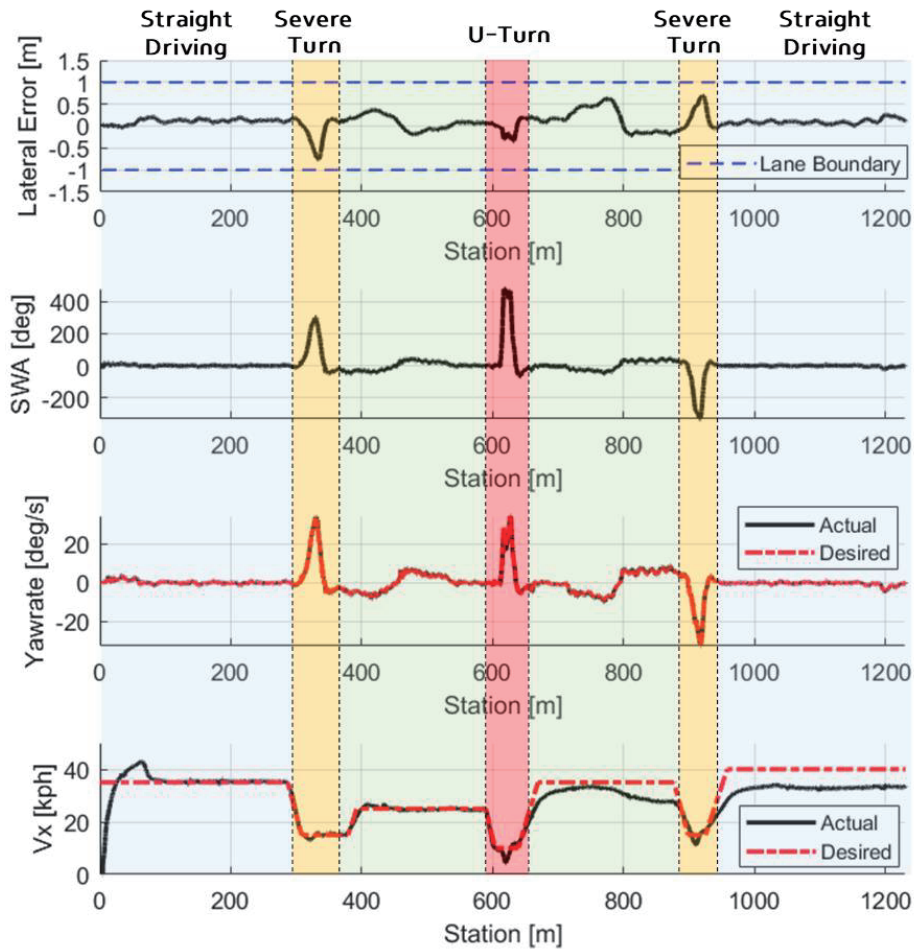

Figure 6.14. The desired trajectory tracking performance of the proposed steering controller

The performance of the proposed steering controller has been compared with that of the pure-pursuit steering controller. The comparison has been conducted with the same speed controller for the objective comparison. The comparison test results are presented in Figure 6.15. As shown in Figure 6.15, the two steering controllers have worked with almost the same speed scenario.

Accordingly, the performance comparison test has been conducted with identical conditions, and the test results are compared objectively. The pure-pursuit steering controller kept the test vehicle to stay in the lane boundary except for the U-turn scenario. The lateral deviation of the proposed controller is smaller than that of the pure-pursuit steering controller on average. The remarkable difference in the tracking performance is found in the U-turn scenario. Unlike the proposed steering controller, it seems that abrupt change occurred in the lateral deviation profile of the pure-pursuit steering controller in the U-turn scenario. The abrupt change is caused by the corner-cutting characteristics of the pure-pursuit steering controller.

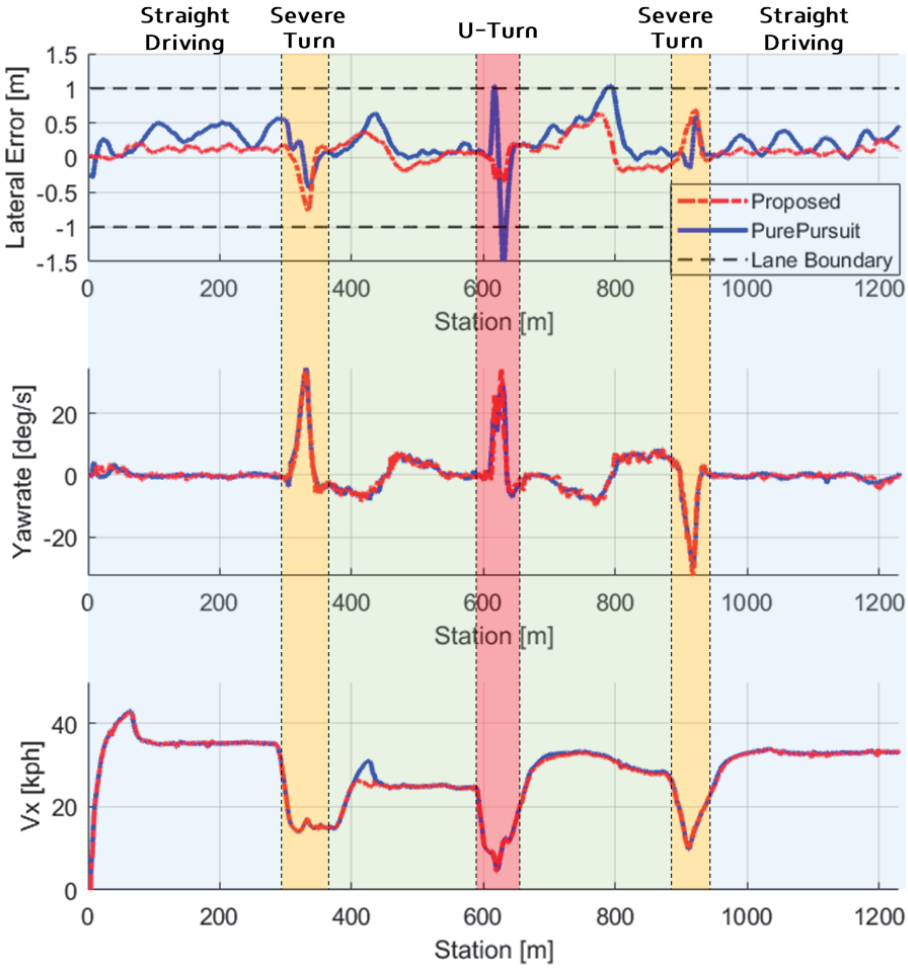

Figure 6.15. The desired trajectory tracking performance comparison between the proposed steering controller and the pure-pursuit steering controller

The driving trajectories of the controllers in the U-turn scenario are compared in Figure 6.16. A solid black line denotes the desired trajectory, and a red dash-dotted line and a solid blue line depict the trajectories of the proposed and the pure-pursuit controllers, respectively. In the figure, the proposed steering controller shows quite better tracking performance in comparison with the pure-pursuit steering controller.

The pure-pursuit steering controller starts to turn early. Corner-cutting behavior makes an inevitable geometrical deviation in the U-turn scenario because the U-turn requires the maximum steering wheel angle. In most cases, the U-turn of a vehicle is allowed in a three-lanes urban road. The more lateral displacement is needed for the U-turn, the more likely to crash to a curb, obviously.

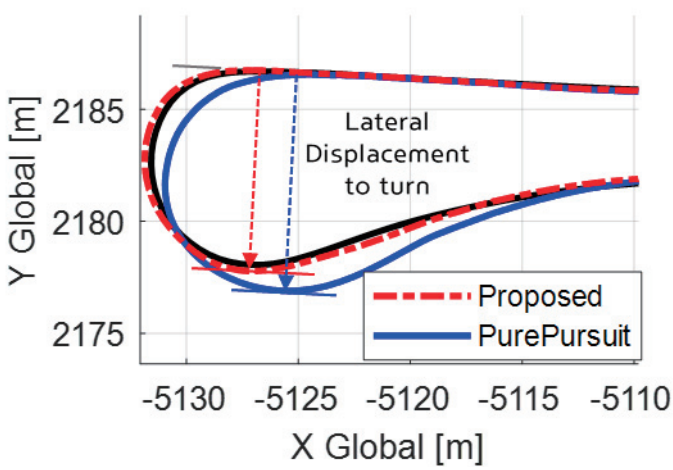

Figure 6.16. The trajectories in U-turn scenario: The proposed controller vs The pure-pursuit controller

As mentioned before, the pure-pursuit steering controller is a quite practical and straightforward approach for the vehicle steering control. The performance of the pure-pursuit approach is directly affected by the selection of a preview target point in the desired trajectory because the desired steering wheel angle is determined by a radius of a circle that links between the current vehicle position and the preview target point. Accordingly, the pure-pursuit steering controller compensates the lateral deviation indirectly. If the target preview point is too close from the vehicle, the chattering of the steering wheel occurs. That leads to degrading the straight course tracking performance. If the target preview point is too far from the vehicle, it would cause severe corner-cutting. Therefore, the preview distance is configured to be proportional to the vehicle speed in most cases. In practical application, the proportional gain to the vehicle speed and the minimum preview distance are considered to be the main parameters.

From the above, the lateral deviation of the pure-pursuit steering controller in the straight driving course, which is shown in Figure 6.15, can be reduced with a longer preview distance.

Figure 6.17 shows the tracking performance of the pure-pursuit steering controller with a longer preview distance. Even though the controller shows quite better performance in the straight driving course, the test vehicle has to be stopped at the left turn corner after the straight driving course, because the lateral deviation exceeded the allowable lateral deviation level.

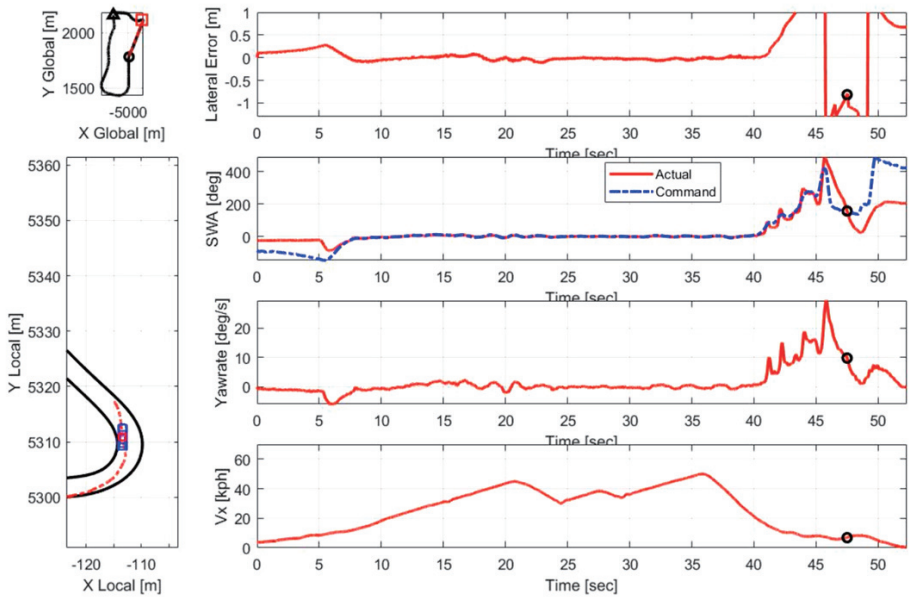

Figure 6.17. The desired trajectory tracking performance of the pure-pursuit steering controller, which is tuned for the near-straight driving course

If the pure-pursuit steering controller in Figure 6.17, is deployed to drive in the near-straight course, it would be a quite good solution. However, it is probably not suitable for the severe curvature course. If the pure-pursuit steering controller is required to handle the severe curvature course, such as U-turn, and right-angle turn in narrow urban road, the degradation of the tracking performance in the straight driving course has to be admitted. That means driving speed is also limited, because the more vehicle speed is increased, the more the vehicle would sweep the lane from side to side.

The tracking performance when the driving speed is increased in the straight driving course, section2 is presented in Figure 6.18. The pure-pursuit steering controller is the same configuration in Figure 6.15. The pure-pursuit steering controller for handling the U-turn and the straight driving course in Figure 6.15 shows poor tracking performance, when the vehicle speed is increased.

On the other hand, the proposed steering controller maintained the performance regardless of the vehicle speed. From the vehicle test results, the pure-pursuit steering controller seems to need the gain-scheduling of tuning parameter for every specific driving scenario.

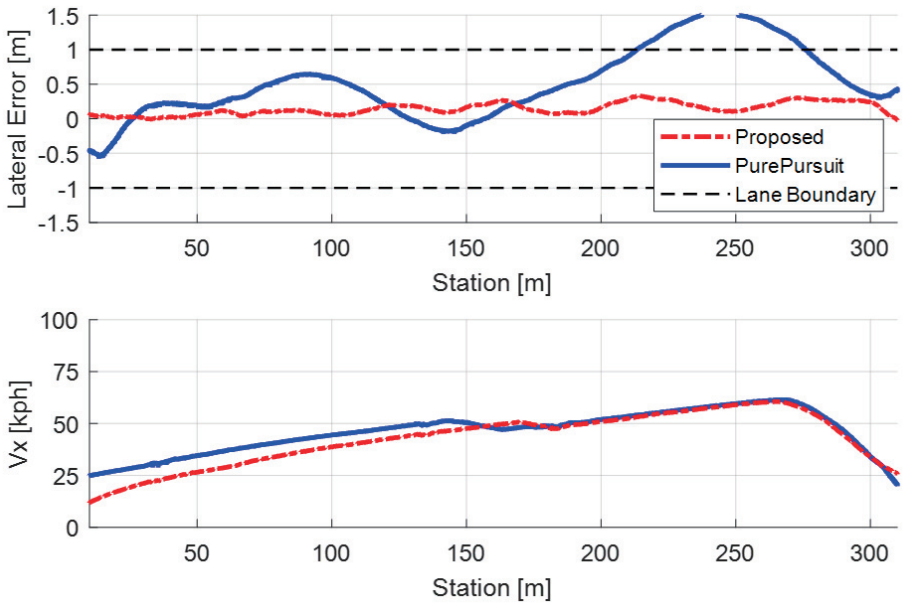

Figure 6.18. The tracking performance with high-speed driving in straight driving course: The proposed controller vs The pure-pursuit controller

The difficulty of tuning in the pure-pursuit steering control for the general driving case is caused by the main concept of the controller. The pure-pursuit controller compensates the lateral deviation indirectly because the controller calculates the desired curvature of a circular arc, which links between the current position of the vehicle to the target point in the desired path. If the vehicle drives in the near-straight course, the lateral deviation can be compensated from the above strategy. However, if the pure-pursuit controller is required to handle the severe curvature, such as U-turn, the preview distance has to be adapted. The indirect method for lateral error compensation causes difficulties to find a general solution.

In contrast with the pure-pursuit steering controller, the proposed controller is not only able to handle the U-turn scenario but also secures stable performance in the near-straight driving. The proposed steering controller prevents excessive steering command, unlike the pure-pursuit steering controller. Accordingly, there is no fluctuation in steering wheel behavior.

As one of the main purposes of this research, the proposed steering controller is developed to cover the general driving scenario. The purpose of the proposed steering controller has been validated via vehicle test results.

Chapter 7

Conclusions and Future Works

This dissertation has proposed a steering control algorithm of the automated driving vehicle for the complex-urban scenario, including U-turn.

The proposed steering control algorithm utilizes the future driving trajectory prediction. The future driving trajectory is predicted using the online learned the vehicle model. The vehicle model is identified online with the streaming data of the current vehicle states.

The Incremental Sparse Spectrum Gaussian Process Regression (ISSGPR) approach is deployed for the online vehicle model learning without the prior knowledge of the dynamics characteristic of the target vehicle. Accordingly, the online learning approach can reduce the required time for applying different types of vehicles.

The proposed steering controller compensates for the current and predicted tracking error both. The predicted tracking error is calculated from the predicted future driving trajectory. The predicted future driving trajectory is similar to the parking guidelines in the commercialized driver assistant system, and the idea

of the predicted error compensation is quite effective to the excessive/deficient steering input for the arbitrary trajectory tracking.

The tracking performance of the proposed steering controller has been validated from the vehicle tests and simulations with challenging scenarios, including the U-turn and the straight course.

The proposed steering controller is compared with the pure-pursuit and the model predictive control approach in terms of tracking performance and the computation load both. The proposed steering controller shows better performance than the conventional methods for trajectory tracking.

For future work, the verification in more various driving conditions, including the snowy and icy road, should be conducted for the reliability of the proposed steering controller. The automated driving system has to work regardless of the weather conditions. Hence, the robust performance of the steering controller is crucial for driving safety.

The speed control can also be designed and integrated with the proposed steering controller using the proposed method. The integration between the steering and speed controller is essential for the robust driving performance in the general road conditions.

Bibliography

Abe, M. (2015). *Vehicle handling dynamics: theory and application*: Butterworth-Heinemann.

Bochner, S. (1933). Monotone funktionen, stieltjessche integrale und harmonische analyse. *Mathematische Annalen, 108*(1), 378-410.

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., . . . Zhang, J. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

Chan, R. W., Yuen, J. K., Lee, E. W., & Arashpour, M. (2015). Application of Nonlinear-Autoregressive-Exogenous model to predict the hysteretic behaviour of passive control systems. *Engineering Structures, 85*, 1-10.

Committee, S. O.-R. A. V. S. (2018). Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *SAE International: Warrendale, PA, USA*.

Droniou, A., Ivaldi, S., Stalph, P., Butz, M., & Sigaud, O. (2012). *Learning velocity kinematics: Experimental comparison of on-line regression algorithms*.

Falcone, P., Borrelli, F., Asgari, J., Tseng, H. E., & Hrovat, D. (2007). Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on Control Systems Technology, 15*(3), 566-580.

Funke, J., Brown, M., Erlien, S. M., & Gerdes, J. C. (2016). Collision avoidance and stabilization for autonomous vehicles in emergency scenarios. *IEEE Transactions on Control Systems Technology, 25*(4), 1204-1216.

Funke, J., & Gerdes, J. C. (2016). Simple clothoid lane change trajectories for automated vehicles incorporating friction constraints. *Journal of Dynamic Systems, Measurement, and Control, 138*(2), 021002.

Gerdes, J. C., & Hedrick, J. K. (1997). Vehicle speed and spacing control via coordinated throttle and brake actuation. *Control Engineering Practice, 5*(11), 1607-1614.

Gijsberts, A., & Metta, G. (2013). Real-time model learning using incremental sparse spectrum gaussian process regression. *Neural Networks, 41*, 59-69.

Goh, J. Y., & Gerdes, J. C. (2016). *Simultaneous stabilization and tracking of basic automobile drifting trajectories.* Paper presented at the 2016 IEEE Intelligent Vehicles Symposium (IV).

Guo, L., Ge, P.-s., Yang, X.-l., & Li, B. (2014). *Intelligent vehicle trajectory tracking based on neural networks sliding mode control.* Paper presented at the Proceedings 2014 International Conference on Informative and Cybernetics for Computational Social Systems (ICCSS).

Hong, S., Lee, C., Borrelli, F., & Hedrick, J. K. (2014). A novel approach for vehicle inertial parameter identification using a dual Kalman filter. *IEEE Transactions on Intelligent Transportation Systems, 16*(1), 151-161.

Jung, C., Kim, H., Son, Y., Lee, K., & Yi, K. (2014). *Parameter adaptive steering control for autonomous driving.* Paper presented at the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC).

Kapania, N. R., Subosits, J., & Gerdes, J. C. (2016). A sequential two-step algorithm for fast generation of vehicle racing trajectories. *Journal of Dynamic Systems, Measurement, and Control, 138*(9), 091005.

Kim, H., Kim, D., Shu, I., & Yi, K. (2015). Time-varying parameter adaptive vehicle speed control. *IEEE Transactions on Vehicular Technology, 65*(2), 581-588.

Kritayakirana, K., & Gerdes, J. C. (2012). Using the centre of percussion to design a steering controller for an autonomous race car. *Vehicle System Dynamics, 50*(sup1), 33-51.

Laurense, V. A., Goh, J. Y., & Gerdes, J. C. (2017). *Path-tracking for autonomous vehicles at the limit of friction.* Paper presented at the 2017 American Control Conference (ACC).

Lee, C., Hedrick, K., & Yi, K. (2004). Real-time slip-based estimation of maximum tire-road friction coefficient. *IEEE/ASME Transactions on mechatronics, 9*(2), 454-458.

M'ng, J. C. P., & Mehralizadeh, M. (2016). Forecasting East Asian indices futures via a novel hybrid of wavelet-PCA denoising and artificial neural network models. *PloS one, 11*(6), e0156338.

Navigant-research. (2019). Navigant research leaderboard report: Automated driving vehicles. In *Assessment of strategy and execution for 20 Companies Developing Automated Driving Systems*.

Rahimi, A., & Recht, B. (2008a). *Random features for large-scale kernel machines.* Paper presented at the Advances in neural information processing systems.

Rahimi, A., & Recht, B. (2008b). *Uniform approximation of functions with random bases.* Paper presented at the 2008 46th Annual Allerton Conference on Communication, Control, and Computing.

Rajamani, R. (2011). *Vehicle dynamics and control*: Springer Science & Business Media.

Rasmussen, C. E. (2003). *Gaussian processes in machine learning.* Paper presented at the Summer School on Machine Learning.

Sanketi, P. R., Zavala, J. C., & Hedrick, J. K. (2005). Dynamic surface control of engine exhaust hydrocarbons and catalyst temperature for reduced coldstart emissions. *IFAC Proceedings Volumes, 38*(1), 206-211.

Song, B., Hedrick, J. K., & Kang, Y. (2014). Dynamic surface control and its application to lateral vehicle control. *Mathematical Problems in Engineering, 2014*.

Stepanyan, V., & Krishnakumar, K. (2010). *MRAC revisited: Guaranteed performance with reference model modification.* Paper presented at the Proceedings of the 2010 American Control Conference.

Subosits, J. K., & Gerdes, J. C. (2017). *A synthetic input approach to slip angle based steering control for autonomous vehicles.* Paper presented at the 2017 American Control Conference (ACC).

Subosits, J. K., & Gerdes, J. C. (2019). From the Racetrack to the Road: Real-Time Trajectory Replanning for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles, 4*(2), 309-320.

Suh, J., Chae, H., & Yi, K. (2018). Stochastic model-predictive control for lane change decision of automated driving vehicles. *IEEE Transactions on Vehicular Technology, 67*(6), 4771-4782.

Swaroop, D., Hedrick, J. K., Yip, P. P., & Gerdes, J. C. (2000). Dynamic surface control for a class of nonlinear systems. *IEEE transactions on automatic control, 45*(10), 1893-1899.

Talvala, K. L., Kritayakirana, K., & Gerdes, J. C. (2011). Pushing the limits: From lanekeeping to autonomous racing. *Annual Reviews in Control, 35*(1), 137-148.

Theodosis, P. A., & Gerdes, J. C. (2011). *Generating a racing line for an autonomous racecar using professional driving techniques.* Paper presented at the ASME 2011 Dynamic Systems and Control Conference and Bath/ASME Symposium on Fluid Power and Motion Control.

Wong, J. Y. (2001). Theory of ground vehicles. John Wiley&Sons. *Inc., New York*.

초 록

가우시안 프로세스 회귀모델을 이용한 실시간 차량 거동 학습 기반 자율주행차량 조향제어

자율 주행 차량은 오늘날 우리 사회가 직면하고있는 다양한 교통 문제에 대한 훌륭한 대안으로 기대를 모으고 있다. 최근의 주행 환경 인지 기술과 실시간 연산 성능의 많은 발전으로 자율 주행 차량이 우리의 생활속으로 빠르게 진입하고 있다. 전 세계 수많은 연구팀이 자율 주행 차량이 대응 가능한 주행 영역을 확장하고 있으며, 정비된 고속도로부터 복합 도심까지 모든 주행 시나리오를 정복하기 위해 지속적으로 발전하고 있다.

복잡한 도심 환경에서 범용적인 주행 시나리오를 대응하기 위해 자율 주행 차량의 조향 제어 기술은 직전부터 U턴과 같은 최소 회전 반경 시나리오까지 안정적인 성능을 확보해야 한다. 따라서 조향 제어 기술의 관점에서 자율 주행 시스템의 주행 영역 확장은 조향각의 사용 범위가 전 영역으로 확대됨을 의미한다.

이러한 이유로, 본 연구에서는 복합 도심 환경 자율 주행을 위한 조향 제어 알고리즘을 개발하는데 중점을 두었으며, 개발된 조향 제어 알고리즘은 직진부터 최소 회전 반경 주행까지 임의의 궤적을 주행할 수 있음을 검증하였다.

제안된 조향 제어 알고리즘은 현재의 경로 오차와 미래의 예측

경로 오차를 모두 보상한다. 미래 예측 경로 오차는 현재 상태로부터 예측된, 차량의 미래 주행 궤적으로부터 결정된다.

미래 주행 궤적의 예측을 위한 차량 모델은 증분형 희소 스펙트럼 가우시안 프로세스 회귀 (ISSGPR) 방법에 의해 실시간 주행 데이터로부터 점진적으로 식별된다. 제안된 실시간 차량 모델 학습 기법은 제어 대상 차량의 거동 특성에 대한 사전 정보가 필요하지 않아 실용성 및 적용성을 확보하였다.

제안된 조향 제어 알고리즘의 성능은 다양한 시나리오에서 진행된 실차 테스트 및 시뮬레이션을 통해 검증되었다. 이로부터, 본 연구에서 제안하는 미래 예측 오차의 보상이 직진에서부터 최소 회전 반경까지 임의의 궤적을 주행하는데 효과적이라는 것을 확인하였다. 또한 제안된 조향 제어기의 성능은 Pure-Pursuit 기법 및 모델 예측 제어 방식과 비교하여 더 나은 추종 성능을 보이는 것으로 확인하였다.

또한 제안된 실시간 차량 모델 학습 접근법이 실시간 주행 데이터로부터 차량의 거동을 예측하는 실용적인 방법임이 검증되었다. 제어기 설계 이전에 선행해야하는 기존의 모델링 파라미터 규명 과정은 시간이 다소 많이 걸리는 작업이었으나, 제안된 기법에선 사전 규명 과정 없이 조향 제어기를 적용 가능하다. 실시간 학습 성능을 정량화하기 위해 ISO / TR 8725:1988의 개루프 조향 입력 시나리오를 사용한 시뮬레이션이 수행되었으며, 다양한 시뮬레이션과 실차 테스트로부터 실시간 학습 차량 모델의 동작이 실제 차량의 동작과 매우 유사함을 검증하였다.

주요어: 자율 주행 시스템, 조향 제어, 궤적 추적, 증분형 희소 스펙트럼 가우시안 프로세스 회귀, 온라인 차량 거동 학습, 합성 입력 기법, 모델 기준 적응식 제어, 유턴 시나리오

학 번: 2013-20640