A NOVEL APPROACH FOR OBJECT DETECTION, PATH PLANNING, AND CONTROL OF AUTONOMOUS VEHICLES ON RURAL ROADS

A Dissertation

by

YANG WOO KIM

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Ivan Damnjanovic
Committee Members,	Alireza Talebpour
	Stephanie Paal
	Dezhen Song
Head of Department,	Zachary Grasley

May 2022

Major Subject: Civil Engineering

ABSTRACT

Many large and small companies have studied and applied autonomous driving systems to the real world. However, much of it is still lacking. In fact, it is easy to find news of accidents of vehicles with autonomous driving technology. Research related to autonomous driving is directly related to human life, so no error should be allowed, and much research effort is needed yet. In order to meet the three significant aspects of current self-driving research: driving accuracy, safety, and comfort, we have constructed three topics that can cover all four essential elements of self-driving cars, which are perception, path planning, decision making, and controller. In this research, we will propose the novel approached methods to solve the current issues of autonomous vehicles and discuss the simulation results which were conducted through our newly implemented simulation environment. Through this, we will show our novel methods have acceptable performance and have breakthrough ideas that others didn't consider yet that can contribute to the future autonomous driving research area.

ACKNOWLEDGEMENTS

First of all, I would like to express my infinite thanks to my committee chair, Dr. Ivan Damnjanovic, for guiding me through endless support and valuable advice. Sometimes you showed me the true scholar as a great professor, and sometimes led me with sincere and warm consideration like my father, so I was able to successfully endure this long journey. I would also like to thank Dr. Alireza Talebpour, my second advisor, for letting me take the first step into my current research field and supporting me with great advice. And I also thank Dr. Stephanie Paal, and Dr. Dezhen Song for accepting to be my committee member and advising me with great suggestions for my research until the end.

This journey was not possible without helps of my wife Woori, who trusted me and endured the difficult 6 year journey with me, my son Daniel, born in 2020 during this journey and brought great joy and happiness, my father and mother, who have been the biggest supporter for their son going for a Ph.D. degree at a late age, and my brother-in-law and sister who always cheered me not to lose confidence. And all my friends and team members who I cannot listing all here. Thank you and I love you so much!

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professor Ivan Damnjanovic, Alireza Talebpour, and Stephanie Paal of the Department of Civil & Environmental Engineering and Professor Dezhen Song of the Department of Computer Science & Engineering.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by a research fellowship from Texas A&M University.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

1.1. Autonomous Vehicles

An autonomous vehicle (a self-driving car) is a motor vehicle capable of operating on its own without the operation of a driver or passenger. The concept of autonomous vehicles was proposed around Mercedes in the 1960s, and rudimentary research began in the mid to late 1970s. Initially, it did not cross the centerline or lane at test driving sites where there were no obstacles, but in the 1990s, autonomous driving areas where obstacles were involved began to be studied in earnest as computer judgment technology developed significantly. Furthermore, research on autonomous driving technologies using deep learning has progressed rapidly in the 2010s and has been limited to commercial vehicles. Currently, tech giants, startups, and automakers are making large investments in the development of self-driving cars, and prototype self-driving cars are being tested on public roads and highways around the world, including the United States, Canada, Europe, Korea, and Japan. In line with companies' development efforts, the state of California passed a bill on February 26, 2018, to allow self-driving cars to be tested on public roads without human drivers on board. Self-driving cars, no longer the technology of the future, are preparing for a tremendous change in the ecosystem of the traditional automobile industry, as well as the structure of the human lifestyle and service industry, as well as the design and national policy of the entire city.

Figure 1.1 Changes in the computing machine of the autonomous vehicle over time.

In March 2004, self-driving cars were developed in earnest, starting with the unmanned car race held in the Mojave Desert under the auspices of America's main military-research Agency (DARPA). Researchers from Stanford and Carnegie Mellon University, who participated in the race, are leading self-driving car projects by Uber, Tesla, and startup companies, including the autonomous car research institute established by Google in 2009. With innovation in the sensor sector in 2005, computer vision capabilities have improved rapidly. It also achieved great results in the field of artificial intelligence systems through deep learning using big data. Currently, prototype self-driving cars are rapidly improving performance with advances in improved sensor processing technology, adaptive algorithms, high-resolution maps, Vehicle to Vehicle (V2V), and Vehicle to Infrastructure (V2I) communication technologies. Technically, autonomous driving in a controlled environment may seem to be already in its final stages.

In autonomous vehicles, it is important to collect various information that exists around cars because cars, not people, have to make decisions necessary to drive. Based on this information, the car makes its own judgment and drives itself. In self-driving cars, sensors will be able to collect information from people around them. Sensors are

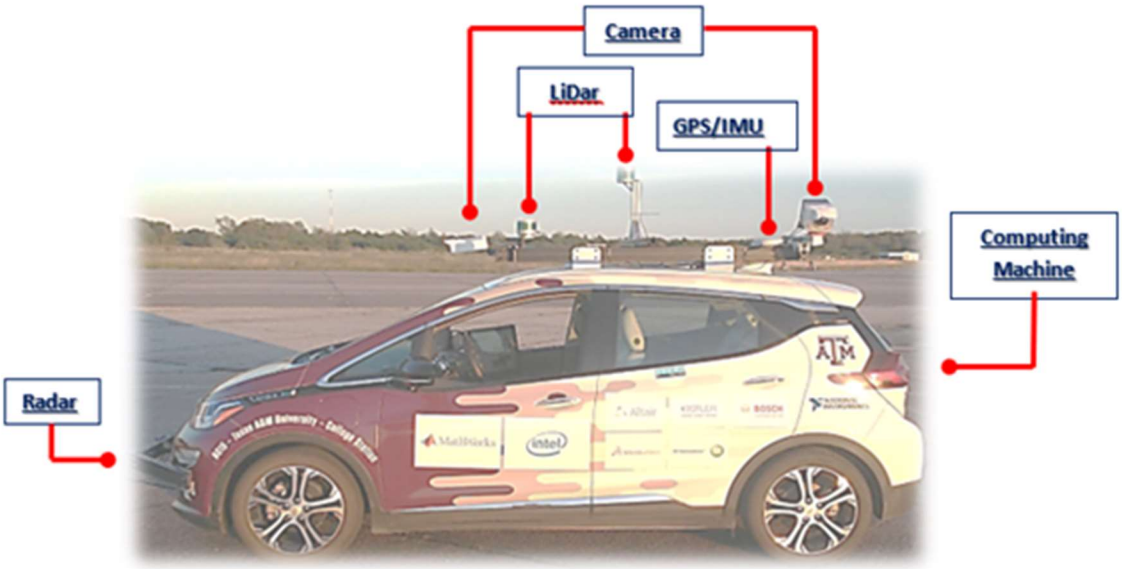


Figure 1.2 Sensors for autonomous vehicle

technologies that replace human vision and hearing and are types of sensors that are installed in vehicles with various cameras, radar, lidar, and ultrasound. Sensors installed inside the vehicle are divided into optical-based sensors and non-optical-based sensors. Optical-based sensors are mainly used to recognize topography and distance, including cameras and laser scanners (LiDARs). Non-optical-based sensors are often used for distance measurements, including radar and ultrasound. Self-driving cars, like humans, collect information in response to dynamic driving environments, perform various algorithms to make judgments about situations, and formulate strategies. These strategies are quickly adapted to changes in the driving environment and are carried out through vehicle control such as steering, acceleration, and deceleration. Thus, the technology of self-driving cars can be classified around cognitive, judgmental, and control functions, all of which operate error-free, enabling stable driving, with artificial intelligence systems and software playing a pivotal role.

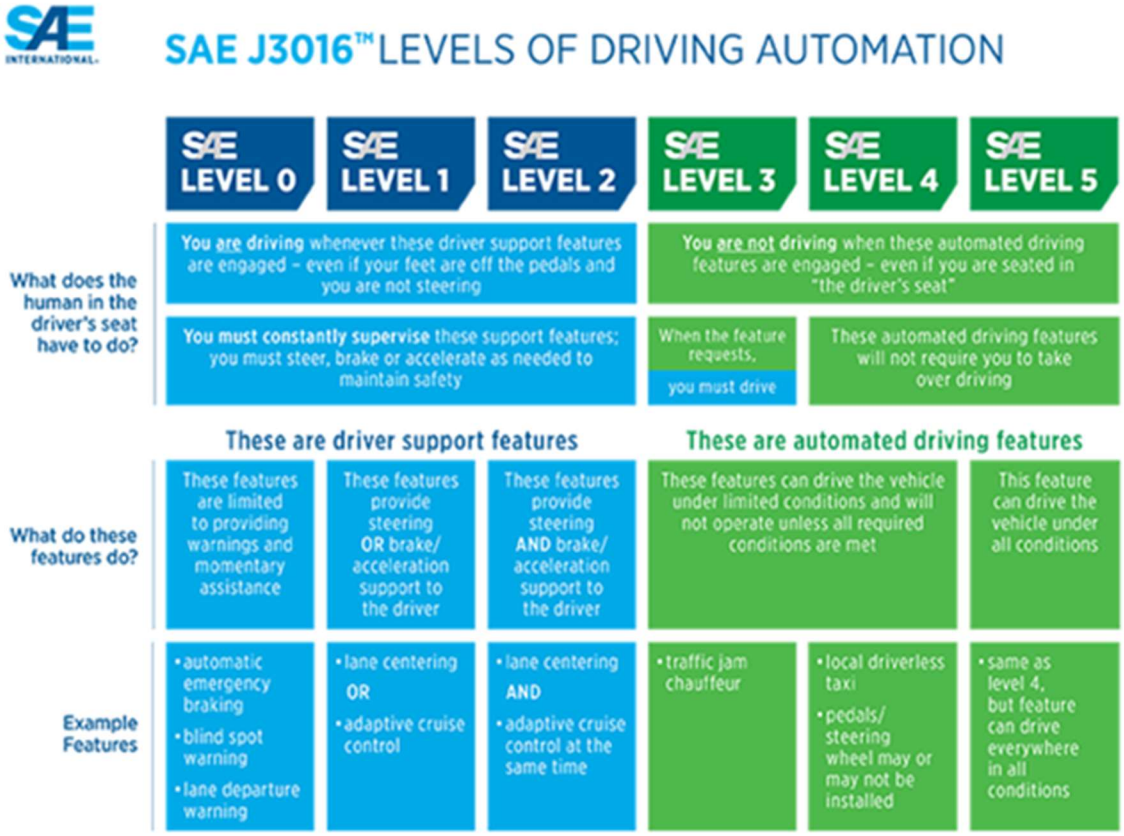


Figure 1.3 SAE J3016 Levels of driving automation. Reprinted from [1]

The Society of Automotive Engineers (SAE) provided a six-step guide to automation levels for self-driving cars in 2014. [1] This distinction is now widely used by officials throughout the self-driving car industry. Phase 0 is the level at which automation-related cars are not applied. Phase 1 refers to the level of motor vehicles currently applied with driver assistance technologies. Phase 2 is a partial automation phase, which is the level of automation with an advanced Auto Driver Assistant System (ADAS) technology. Phase 3 is a stage of self-driving under limited conditions, in which drivers must be always on standby in case of unexpected situations. Phase 4 is the highest automation phase that does not require any devices needed for human driving,

and there is a limit to driving only in certain areas and at certain speeds. Phase 5 is a fully automated stage, which means automation that is beyond the limits of region and speed while humans do not have to pay any attention to the driving of cars. Currently, the technology level of self-driving cars is in the process of moving from phase 3 to phase 4, and it is believed that only when the automation of phase 4 and 5 takes place will it enter the technology level that can provide the aforementioned benefits to us.

Table 1.1 2020 2020 Disengagement Reports, State of California DMV, Autonomous vehicle manufacturers that are testing vehicles in the Autonomous Vehicle Tester (AVT) Program and AVT Driverless Program are required to submit annual reports to share how often their vehicles disengaged from autonomous mode during tests (whether because of technology failure or situations requiring the test driver/operator to take manual control of the vehicle to operate safely).

Company	Number of tested vehicles	Total Driven Mileage	Total Disengagement	Disengagement per 1,000mi
Waymo LLC	239	628,838.50	21	**0.03**
CRUISE LLC	137	770,049.26	27	**0.04**
AutoX Technologies, Inc	8	40,734.00	2	**0.05**
PONY.AI, INC.	29	225,496.00	21	**0.09**
WeRide Corp	9	13,014.00	2	**0.15**
DiDi Research America LLC	12	10,401.49	2	**0.19**
Nuro, Inc	20	55,369.84	11	**0.20**
Zoox, Inc	45	102,521.00	63	**0.61**
Aurora Innovation, Inc.	12	12,200.78	37	**3.03**
Lyft	19	32,731.36	123	**3.76**
Gatik AI Inc.	3	2,352.00	11	**4.68**
Apple Inc.	69	18,805.30	130	**6.91**
Nissan North America, INC	4	394.50	4	**10.14**
BMW of North America	5	122.00	3	**24.59**
AImotive Inc.	3	2,987.00	113	**37.83**
Mercedes Benz R&D North America, Inc	10	29,983.80	1167	**38.92**
NVIDIA	7	3,033.00	125	**41.21**
QUALCOMM TECHNOLOGIES, INC.	3	1,727.00	90	**52.11**
SF Motors, Inc.	2	874.69	61	**69.74**
EasyMile	1	424.00	128	**301.89**
Toyota Research Institute	7	2,875.00	1215	**422.61**
Telenav, Inc.	1	4.00	2	**500.00**
Udelv, Inc	2	66.00	49	**742.42**
Ridecell Inc	1	147.63	189	**1280.23**
Valeo North America Inc.	2	49.00	99	**2020.41**

Many large and small companies have studied and applied autonomous driving systems to the real world. Many people think that self-driving technology research has been conducted a lot, but in reality, much of it is still lacking. In fact, it is easy to find news of accidents of vehicles with autonomous driving technology. And, according to the Autonomous vehicles' disengagement report provided by the California DMV [2], many self-driving car companies test their self-driving cars and see how often the self-driving mode is disengaged by various factors. The indicators also show that research on autonomous driving has not yet been completed and that there are many studies left to be carried out in the future. As research related to autonomous driving is absolutely directly related to human life, it is a very important and large part of our human future, so no error should be allowed, and much research effort is needed yet.

In this research, we would like to introduce a number of ways to achieve great progress in the perfection of autonomous vehicle systems directly connected to human life, as mentioned earlier. To this end, this paper seeks to contribute to perfection in three aspects: 1. Safety, 2. Driving accuracy, and 3. Driving comfort.

1.2. Objectives of this study

This research consists largely of three parts, which cover all four of these sub-modules of driverless automotive systems while also satisfying Safety, Driving Accuracy, and Driving Comfort.

1.2.1. Perception

It is a step in which data from cognitive sensors such as cameras, LiDAR, Radar, etc. are used to obtain information about lanes and surrounding objects. This is a module that acts as a person's vision, hearing, or touch.

1.2.2. Path Planning

It is a stage in which all paths for a vehicle to be driven are pre-planned using information passed by the perception module. All paths that the vehicle can travel will be created, and one of these paths can be selected in the next step, the decision maker.

1.2.3. Decision maker

It is a module that aggregates all the information previously organized in the perception module and path planning module to determine the path and speed of the vehicle. At this point, it determines how quickly or slowly the perceived objects can be avoided in some way. This is exactly the case with the human brain.

1.2.4. Motion controller

It is a module that calculates the steering wheel angle, throttle, and brake values at each moment in order to actually move the vehicle according to the path and speed determined by the decision maker. Calculated values will be transmitted to the vehicle through CAN bus.

1.3. Summary of research topics

This research is a study of self-driving related fields, which still have a lot of deficiencies, divided into following three major topics.

Table 1.2 Research topics

Topics	Descriptions
Topic 1	Early Warnings Signals of Unexpected Path Obstructions for Autonomous Vehicle Controls
Topic 2	Steering Gears Configurations in Pure Pursuit Method for Autonomous Ground Vehicles
Topic 3	Path Optimizations for Autonomous Ground Vehicles Controls

1.3.1. Topic 1

Proposes the early warning based complementary system that would be able to activate collision avoidance control sequence when machine vision algorithms fail to detect and classify objects. The approach is camera-based and utilizes image signal processing algorithms based on the concept of critical slowing down.

1.3.2. Topic 2

Seek to address the controller role when a given path is established from the path planning stage. Introduce a new way to take advantage of the Pure Pursuit Controller, and to make up for the vulnerability with the newly designed Automatic Steering Gears for Pure Pursuit method.

1.3.3. Topic 3

To compensate for the weakness that vehicle controllers are vulnerable to drastic road curvature changes, we propose a new method of smoothing recognized roads. Just like designing the railway path to avoid sharp changes in curvature, we choose to design a path for control of automatic ground vehicles using a transition curve.

1.4. References

[1] SAE International, SAE International Releases Updated Visual Chart for Its "Levels of Driving Automation" Standard for Self-Driving. https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles, Accessed Feb. 20, 2022

[2] State of California DMV, 2020 Disengagement Report. https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/disengagement-reports/

2. COMPLEMENTARY EARLY WARNING SIGNALS FOR OBJECT DETECTION FAILURES OF AUTONOMOUS VEHICLE CONTROLS

2.1. Introduction

On May 6, 2016, Joshua Brown, a 40 years old man from Ohio, was killed when his Tesla Model S failed to detect a semi-trailer truck turning across its path and collided with the side of the truck [1]. At the time of the accident the vehicle was operated by Autopilot, a computer-assist mode available on the Model S. Even though Tesla Motors was ultimately cleared of any wrongdoing, the crash has since threatened to sidetrack the whole industry effort for advancing vehicle automation. Specifically, the investigation report from the National Highway Transportation Safety Agency (NHTSA) ironically highlighted that "Autopilot requires full driver engagement at all times!"

Since the fatal incident in 2016, Tesla Autopilot have been involved in two other fatalities in which the driver was killed and the Autopilot mode was confirmed to be engaged at the time of collision – one more fatal collision with turning semi-trailer trucks in Florida, and a fatal collision with a median barrier in California. The first collision occurred on March 23, 2018, in Mountain View, California with a Tesla Model X. The vehicle was following a lead car, and as the path of the lead car veered right, the Model X stopped following and instead accelerated toward a median barrier, resulting in a fatal collision [2]. The second collision occurred on March 1, 2019, in Delray Beach, Florida when a Tesla Model 3 also failed to detect a semi-trailer truck turning across its path and collided with the side of the truck. In this incident, the National Transportation

Safety Board (NTSB) released clear findings that “the car’s sensors weren’t designed to identify the side of the truck and, therefore, didn’t slow the car”.

In addition to the three fatal crashes, some vehicle owners have shared personal stories over social media claiming to have been in collisions caused by a failure of the autopilot system to detect objects. One California driver claims her autonomous vehicle collided with a parked police vehicle while autopilot was engaged. Another Utah driver claims to have collided with a fire hydrant while autopilot was engaged [3]. For its part, Tesla Motors emphasizes in its user manual that collisions with stationary objects while traveling over 50 mph are possible based on current design, so drivers should stay attentive and prepared throughout their drive for these types of scenarios. The Model S user manual states “Traffic-Aware Cruise Control cannot detect all objects and, especially in situations when you are driving over 50 mph (80 km/h), may not brake/decelerate when a vehicle or object is only partially in the driving lane or when a vehicle you are following moves out of your driving path and a stationary or slow-moving vehicle or object is in front of you” (Tesla Model S 2019). It is apparent that the current object detection methods are capable of failure, and there is room for improvements to be made.

In fact, there is a fundamental underlying issue at the root of the problem. When vehicles travel at higher speeds relevant driving environment extends beyond what LiDAR and long-range radar sensors can effectively perceive in presence of noisy backgrounds. Cameras can see further, but the obtained images are not well suited for

training machine vision object detection and classification algorithms. This is particularly problematic for objects that laterally enter the field of camera's vision.

Geirhos at al. [4] carried out comparison of human vs. deep neural networks (DNNs) object recognition robustness for image degradations. According to their study, DNN is extremely vulnerable to small changes in the operating environment and image file corruption, as well as DNN is very easily affected by random pixel disturbances compared to humans. The performance of DNN reduced rapidly as noise and distortion increased, compared with human cognitive abilities.

In fact, in autonomous driving applications image distortions occur very frequently, especially during driving in high speeds. This image distortion and the lack of robustness represents a serious limitation of object detection technology, which is now widely used for autonomous driving. More accurate training data can perhaps improve the detection, but these data are not easy to collect in real driving situations.

Driving environment is dynamic and prone to constant changes; a driver may expect vehicles to cross his/her path in a variety of ways for a variety of reasons, many of which may be unexpected. Human attention to potential hazards is triggered by early warning signals; the events that indicate tipping points after which the environment is expected to suddenly change. In context of data analysis, these tipping points can be any number of statistical indicators; critical points, sharp changes in mean or variance values or their differentials, the introduction of new system elements, or other signals that the environment is on the brink of change. Early warning signals provide an additional level

of security to the driving environment, predicting collision events far enough in advance that corrective action can be taken.

In this paper we hypothesize that the risk assessment methods based on the concept of critical slowing down when applied to image processing could improve the set of early warning signals available to autonomous vehicles to detect impending vehicle collisions, more specifically unexpected occurrence of lateral obstruction objects within the relevant driving environment. The early warnings method is presented in the Methodology section of this paper, with the results of the experimental analysis following.

2.2. Background

In autonomous driving numerous sensors provide the data inputs necessary for the vehicle to sense its surrounding environment. One of the primary means by which autonomous vehicles sense their environments is by object detection and tracking.

A common method of image detection used for machine vision is a convolutional neural network, or CNN, or a region convolutional neural network, or R-CNN. These methods detect objects and draw boundary boxes around detected objects to localize the object within the source image [5]. R-CNN, as well as its variants Fast R-CNN and Faster R-CNN, is regarded as one of the most efficient means of object detection. R-CNN generates 2,000 regions within an image to search using selective search algorithms. From there, CNN methods are applied to extract features and classify object presences. R-CNN has a speed advantage over CNN due to its limited number of regions

to search, but it is still a relatively slow method of object detection. Fast R-CNN flips the first two steps of the R-CNN process; instead of first generating regions and then feeding them to a CNN processor, the fast R-CNN process feeds a source image to a CNN processor to form a convolutional feature map, and then uses that feature map to identify proposal regions. Faster R-CNN works similarly to fast R-CNN, but rather than finding region proposals through selective search, the program uses machine learning to learn how to predict region proposals. As a result, this method can detect objects in as little as 0.2 seconds, making it a realistic method to use for real-time driving object detection [5][6][7].

Lefèvre et al [8] provide a survey of the existing methods of motion prediction and risk assessment for intelligent vehicles in 2014. The survey classified three types of motion models utilized by autonomous vehicles to predict the behavior of other drivers and avoid collisions: physics-based models, maneuver-based models, and interaction-aware models. However, in all three types of models one assumes that the object is identified.

Physics-based models have been studied extensively, and they are some of the most common models used today for collision risk estimation. However, they are simplistic models which can predict motion behavior no more than a second in advance, and they cannot predict changes in vehicle motion as a result of maneuvers or external factors [8][9][10]. Maneuver-based models build off of physics-based models by considering the potential future motions of a vehicle based on the maneuvers that a driver intends to perform, independent of maneuvers performed by other vehicles [8]. It

considers changes in vehicle behavior as a result of maneuvers, such as a change in direction or slowing down as a result of a turn [11][12].

Maneuver-based models suffer limitations for their assumption that vehicles operate independently. Instead, vehicles and drivers interact with one another on the roadway, and one driver's maneuvers will impact another's [13][14]. Ignoring these effects results in inaccurate risk estimations. Interaction-aware models resolve the deficiencies in the maneuver-based models, considering the inter-dependencies between vehicle maneuvers.

A downside of object detection and following path prediction is the extensive training is needed to work effectively. Furthermore, some objects may fall get detected at all, like the side profiles of semi-trailer trucks as noted in two fatal autonomous vehicle collisions. An analysis of images and pixel behavior in a vehicle's camera video feed can serve as a supplemental strategy to reduce the likelihood of collisions occurring.

Pixel behavior can be studied for early warning signals - changes in statistical data which indicate that a system is at a tipping point of change. These changes in behavior can be changes in mean grayscale values, sudden changes in variance values, changes in moving averages through video clips, or other abnormalities which indicate a possible change in the system environment. Since observations about pixel behavior provide object detection information at best, with no object classification data and little localization data.

A 2008 study examining vehicle detection based on traffic surveillance videos examined data similar to that collected for this research - rectilinear stationary camera footage taken on a roadside rather than mounted in-vehicle. The data collected for this study was intended for use in traffic counting and regulation, comparing results between a wide-lens camera and inductive loops. The results found that even with a simplistic pixel-based algorithm, the camera was able to outperform the loop detectors for vehicle detection [15].

One popular technique of using pixel behavior to identify moving objects involves modeling each individual pixel with a Gaussian distribution. Gaussian techniques allow for a frame-by-frame subtraction; changes between frames will be noticeable as non-zero values. A 1997 review of a people-tracking system utilized a single Gaussian model in this manner [16]; however, this system did not perform well in environments with dynamic backgrounds or scenes.

Gaussian techniques are not always appropriate for pixel behavior analysis. A 2002 study proposed a non-parametric model for analyzing pixel densities; pixel intensity probability density functions are assumed to vary from image to image and from frame to frame without an assumed parametric form. Instead, the probability density functions are estimated at directly at the data-level, using kernel density estimation techniques, without any underlying assumptions about the form of the distribution [17].

Another example of utilizing pixel variations to identify objects is using discriminative texture features to reduce the effects of background events. Backgrounds

can pose challenges to object detection since they are often dynamic; trees move, water flows, birds fly through the frame, and shadows are cast around. Cameras can sway or bounce, and some cameras auto-adjust for light levels, causing changes to the image. Texture operators can be utilized to detect and isolate background pixel variations so that objects can be detected with fewer false positives [18].

Sometimes, early warning detectors will provide false positives - instances where early warnings are triggered, but no transition event occurs. False positives are easily identifiable in historical data since one can simply compare the instances of warning signals to the actual events which have occurred. In real-time or future data however, it is nearly impossible to distinguish a false positive from a true positive [19]. Some false positives occur due to random coincidences, while others are unintentionally systematically built into the early warning system framework. The systematic false positives are problematic, but they can be reduced or removed from the system through various treatments. One such treatment involves running the early warning system against a system with no critical transitions; therefore, any detected positive is a false positive, which can be investigated for root cause. Another treatment involves using local or global scaling to eliminate the influence of original data points [19]. Many additional treatments exist as well. Early warning systems are a delicate balancing act between preventing false negatives and avoiding false positives.

2.3. Methodology: Image-based early warning signals

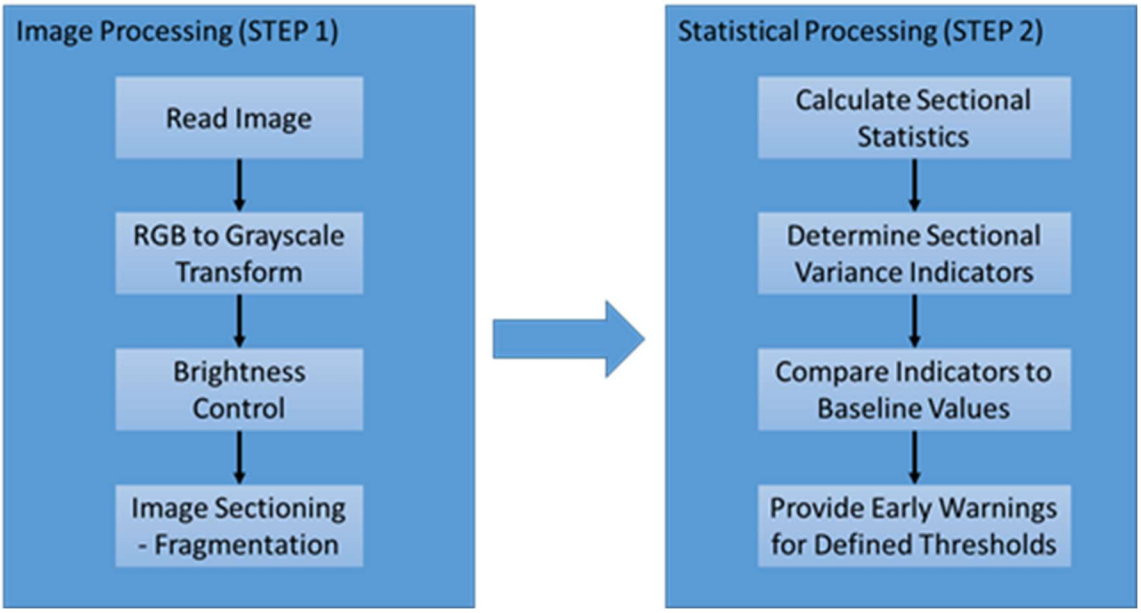


Figure 2.1 EWS Framework

The overall early warning system covered in this paper can be largely divided into two sequential steps. The objective of the first step - Image Processing is to remove unnecessary information and noise from the images, while the objective of the second step - Statistical Processing is to process the image pixels and determine its behavior using the concept of critical slowing down and time series analysis. Figure 2.1 shows the overall process that encompasses the previously mentioned two steps.

2.3.1. Image processing

The images captured in vehicle's camera have more detailed information than one needs to detect early warning signals. The original data is a high-resolution color-based image frame that is difficult to process in real-time. To address this issue one can convert it into a gray scale image to significantly reduce the amount of data processed.

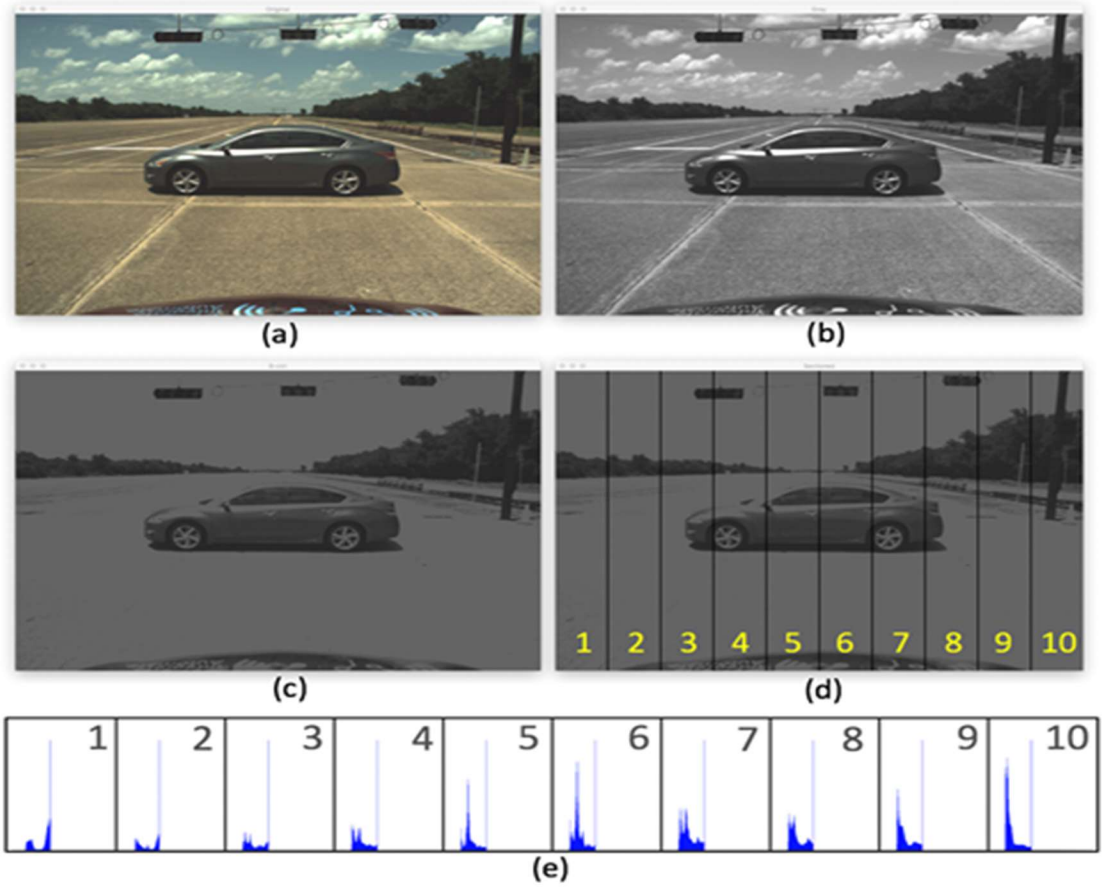


Figure 2.2 Image Processing. Original(a), Gray scaled(b), Brightness controlled(c), Sectioned(d), and 0~255 gray scaled pixel distribution for each section(e).

Compared to the processing of three channels in RGB, it is much simpler and faster to process data on only one gray scaled channel with values between 0 and 255. To convert three channels RGB image to grayscale one can use the following formula.

$$Gray(x,y) = 0.299 * Red(x,y) + 0.587 * Green(x,y) + 0.114 * Blue(x,y) \qquad (2.1)$$

In addition, brightness control can be added to better detect moving objects and eliminate unnecessary noise. In this paper, a simple brightness control method was used.

There were many unnecessary bright pixels in the sample video we used in the experiment, so we set the maximum pixel value to remove the bright spots.

$$Dst(x,y) = \begin{cases} Gray(x,y), & if\ Gray(x,y) < 100 \\ 100, & otherwise \end{cases} \quad (2.2)$$

When the desired binary image is ready, the images are divided into several specific sections. In this paper, ten sections were divided vertically because the vertical division allows more effective detection of the position of the vehicle or pedestrian coming close to the left and right from the driver's point of view.

2.3.2. Statistical processing

The statistical processing process can be performed largely in two different ways: 1. directly comparing mean and variance for each section's time sequence and 2. comparing auto/cross-correlation coefficient for each section.

2.3.2.1. Mean and variance method

The critical slowing down phenomenon can be described by considering two different systems – the system that is far from the transition; and the system that is close to the transition. The system characterized by a “deep basin” of attraction with steep slopes around the equilibrium point, subjected to stochastic perturbations, will promptly return to the equilibrium point. In other words, the system is resilient to perturbations. The time series metrics (i.e., variance and autocorrelation of lag 1) will show the

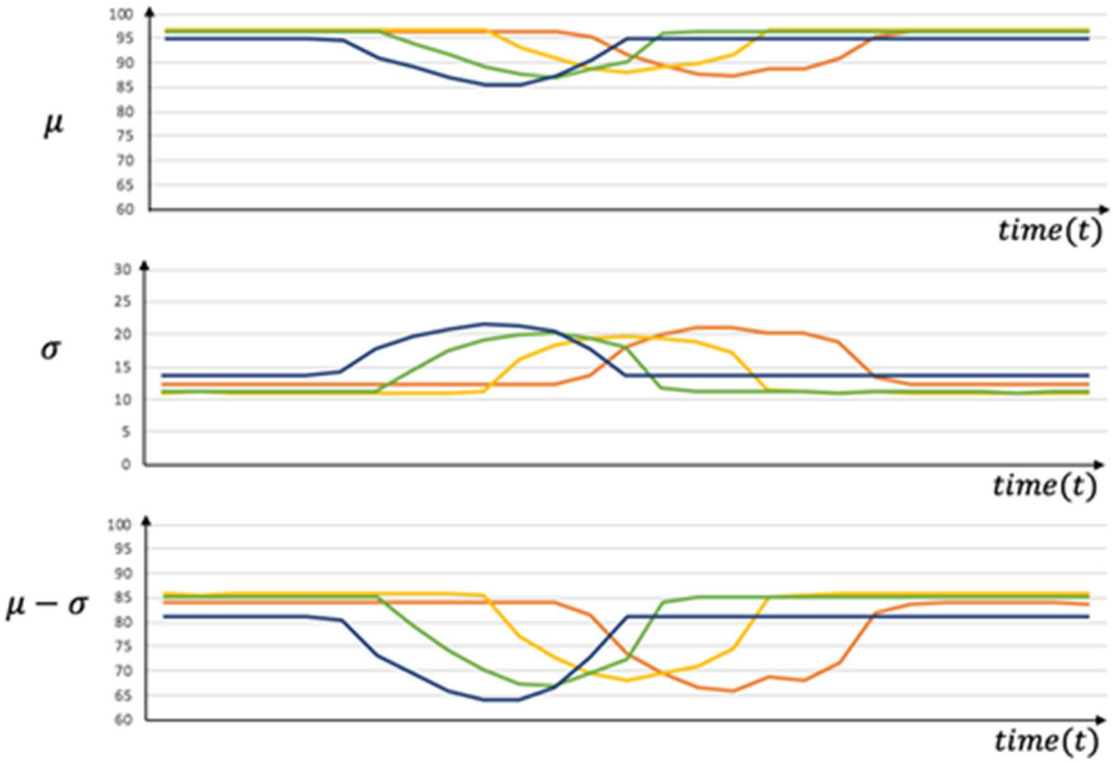


Figure 2.3 Changes in the time of mean(μ), variance(σ), and μ-σ as one vehicle passes by. Four out of ten vertical sections were divided into different colors. As shown in the figure, μ-σ can be used to identify a more obvious change in statistical values.

behavior that is consistent with the system being in a stable orbit; the variance and correlation are constant and relatively low. The system characterized by an equilibrium point with a relatively flat basin of attraction, subjected to stochastic perturbations, will slowly return back to the equilibrium. Hence, even minor disturbances can take the system to the saddle point and into a new basin of attraction that may be associated with fundamentally different system behavior. Far away from the transition point, where the topology is characterized with a “deep basin”, the system exhibits lower autocorrelations and variances; close to transitions, where the topology is characterized with a “shallow basin”, the system shows increases in variances and autocorrelation.

General principles of critical transitioning are universal and have found applications in many disciplines [20]. For example, recovery time models are often used in vibration analysis of rotary equipment to identify changed operating conditions [21], and financial market failure [22]. Similarly, accosting monitoring devices are placed to analyze pressure relief valves and warn against potential failures [23].

In this paper we are concerned with the statistical properties of a signal derived from a video feed i.e., a series of consecutive image frames of the driving environment. The key hypothesis is that the transiency to a new driving environment (i.e., new vehicles appearing and the existing vehicles changing directions) is characterized by the statistical signature of critical slowing down.

The moments of the time series are typical metrics where the statistical signature of critical slowing down appears before transitioning. The variance of the time series is defined as:

$$\sigma^2 = \frac{1}{N-1}\sum_{t=1}^{N}(x_t - \bar{x})^2 \tag{2.3}$$

In this paper we mainly focus on the time series of the mean (μ) and variance (σ) of the pixel of each segment. This is because analyzing signal at the pixel level would be too noisy and difficult to interpret. In fact, Damnjanovic and Avan [24] state that system scale (spatial and time) is one of the key calibrating factors in developing early warning systems.

In the previous step, the whole image is divided into 10 vertical sections, and in this step, the decision value D=μ-σ is obtained after calculating the mean and variance for the pixel value of each section. The reason for μ-σ rather than using the mean directly is that the value of the changed mean is to be magnified through the variation of the image. The final decision value D is calculated by summing D_i for each of these calculated sections. Assuming that there is no moving object in the first frame of the image, the D value of this first frame is set to base value D_b. Once the final D value is obtained, the difference between D and the base value D_b is calculated to determine the current risk level. In this paper, a total of five risk levels are designed to be known to the driver using four different thresholds.

2.3.2.2. Auto-correlation method

The auto-correlations for the time sequence of each of the 10 sections can be calculated to identify the changes that occur in each section. In addition, the cross-correlations between adjacent sections can be calculated, and the changes in values over time can be identified to capture movement against the surrounding objects. Equation (4) shows the formula used in this paper to calculate the coefficient of coefficient r. The correlation coefficient is always between -1 and +1, where -1 represents a negative correlation between X and Y, while +1 represents a positive correlation between X and Y.

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \tag{2.4}$$

Figure 2.4 ROI when the controlled vehicle is moving. The original frame (top) and clipped frame by ROI (bottom).

As mentioned above, auto-correlation coefficient and cross-correlation coefficient are calculated, and for auto-correlation, these values are compared to the values of the previous frames, and for cross-correlation, the values of the adjacent sections are compared to the resulting values of the previous frames. The ability to catch movements in such fine detail is designed to vary depending on the user's set comparative thresholds, th_{auto} and th_{cross}.

2.3.3. Additional suggestions for when the controlled vehicle is moving

What we've covered so far has been a way to catch the movement of objects based on when the vehicle is stationary. However, when a controlled vehicle is moving, all other backgrounds, not just moving objects, may not be very effective in the way introduced so far. To address these issues, this paper establishes region of interest (ROI)

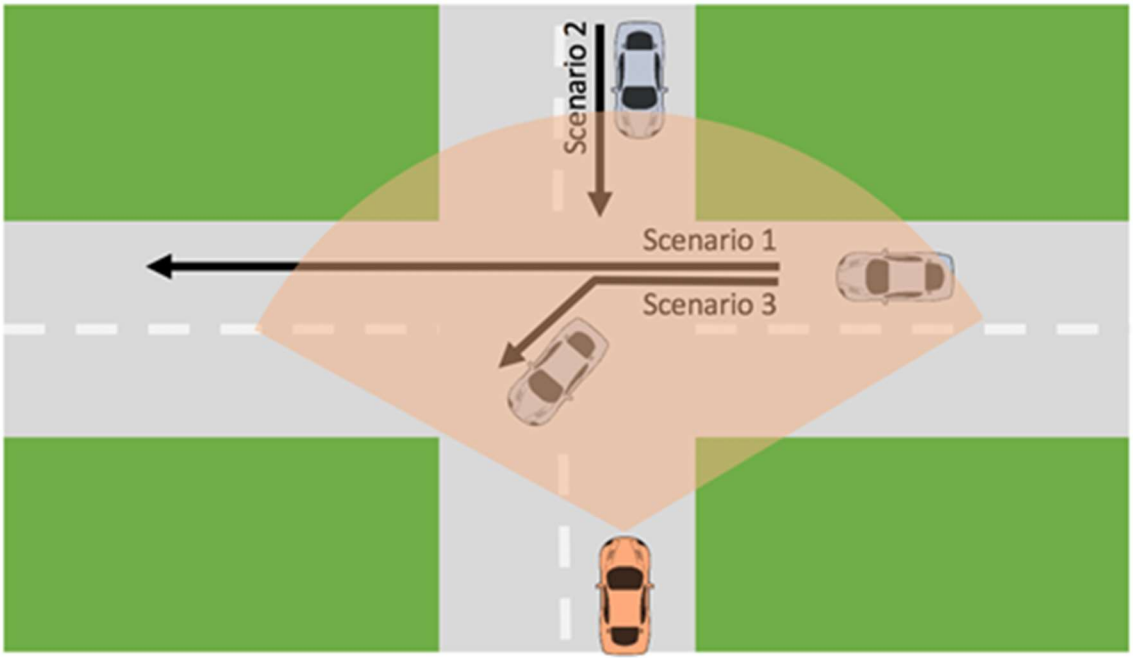


Figure 2.5 Experimental scenarios

to treat only the most critical parts of safety in a statistical manner. The ROI suggested for when the controlled vehicle is moving in this paper is shown in Figure 2.4.

2.4. Experimental setup

The experiments in this study were conducted at Texas A&M University Rellis Campus, where all research experiments on self-driving cars in Texas A&M University is currently under way. For the experiment for this study, FLIR Blackfly S (Model: BFS-PGE-23S3M-C: 2.3 MP, 53 FPS, Sony IMX392, Mono) camera sensors installed on Chevrolet Bolt vehicle from AutoDrive Project (a three-year competition to develop a fully automated vehicle sponsored by General Motors and SAE International) were used. After we stopped the experimental vehicle equipped with the camera at the corresponding intersection position, we prepared the experimental video by video

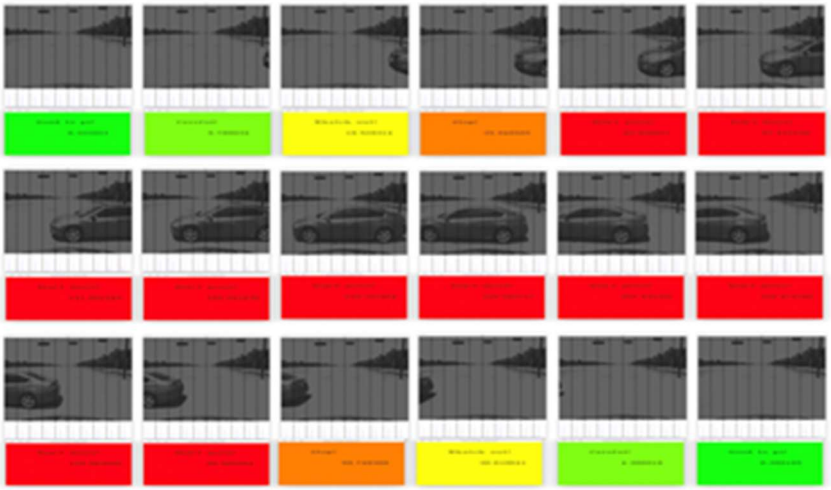
recording the movements of the car ahead and using these prepared videos, we implemented code to handle them. For the implementation, all codes were implemented and tested with C++ and OpenCV(ver.4.0.1) for Image Processing.

Table 2.1 Early warning thresholds

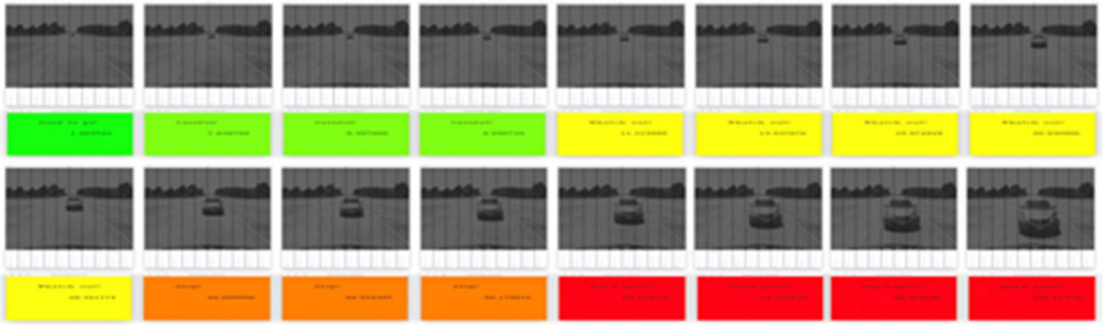
Warning Level	Range	Message	Color
1	|D-Db| <= 2	Good to go!	Green
2	2 < |D-Db| <= 10	Careful!	Light Green
3	10 < |D-Db| <= 35	Watch out!	Yellow
4	35 < |D-Db| <= 60	Stop!	Orange
5	60 < |D-Db|	Don't move!	Red

The practical experiments of this study have been carried out in three cases depending on the movement of the vehicle ahead; 1) if the vehicle is moving horizontally from right to left, 2) if the vehicle is moving longitudinally, 3) if the vehicle passes diagonally ahead. Figure 2.5 illustrates these scenarios. The motivation for focusing on them is the fact that these situations are the ones where the current methods for object identification and tracking show the highest rate of false negatives. The warning threshold for this experiment was set as shown in Table 2.1.

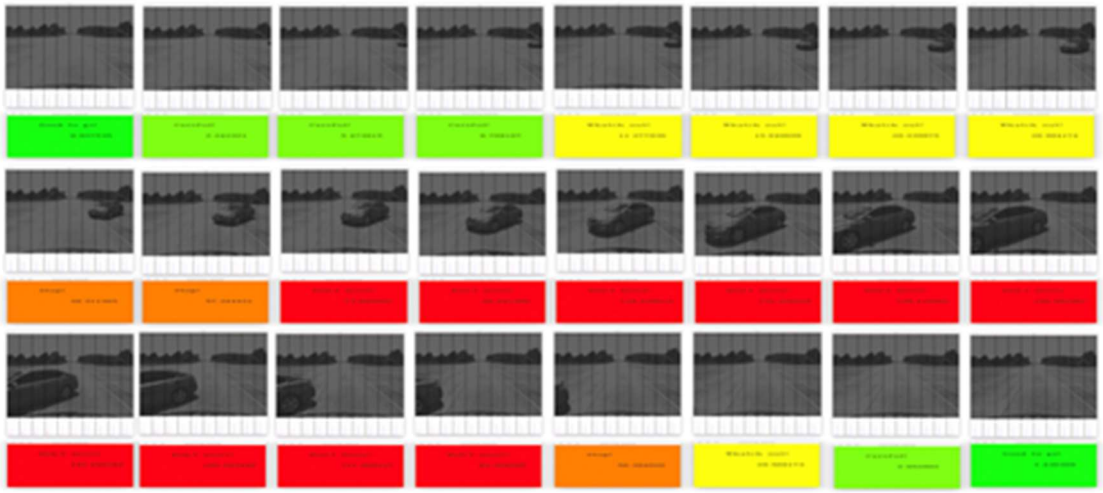
The warning thresholds used in experiments in this paper were determined by calibration through several prior experiments. After dividing the five situations (‘Good to go!’, ‘Careful!’, ‘Watch out!’, ‘Stop!’, and ‘Don’t move!’) that can be felt through the actual driver by the degree of risk, the thresholds were determined to be the ones that



(a) Senario 1



(b) Senario 2



(c) Senario 3

Figure 2.6 Experiment results Scenario 1: Vehicle ahead moving horizontally from right to left(a), Scenario 2: Vehicle ahead moving longitudinally from a distance to a distance(b), Scenario 3: Vehicle ahead passes diagonally ahead(c).

best meet these situations. However, keep in mind that these thresholds can vary depending on the resolution of the camera sensor, the size of the image, and the segmentation method using in Image Processing. For example, in our additional experiment on the dashboard camera footage from Tesla Model S Sedan's accident vehicle in January 2016 in China [25], the video had a low resolution and foggy driving environment, so the threshold was calibrated accordingly.

2.5. Results

Figure 2.6 illustrates the results of the experiments on scenarios listed in the previous section. When a moving vehicle is not in front view, the warning panel displays 'Good to go!' which is the lowest level of warning. If a moving vehicle is very far from the subject vehicle, the calculation for determining the warning level becomes very small and the warning level is also the lowest because of the very low contrast. (Scenario 3) A second warning phase, 'Careful!', is output when the vehicle begins to enter the forward field of vision, and as the vehicle progressively enters the forward field of vision, the warning phase increases.

Since we have divided the sections, controller can recognize the size and location of the hazard. The larger the size of a moving object, the greater the number of sections whose statistics change, and the smaller the object, the smaller the number of sections that change. In addition, information on the left and right positions of moving objects can be obtained depending on the position of the section in which the statistics change. These are the distinctions from simply calculating pixel values for the whole image.

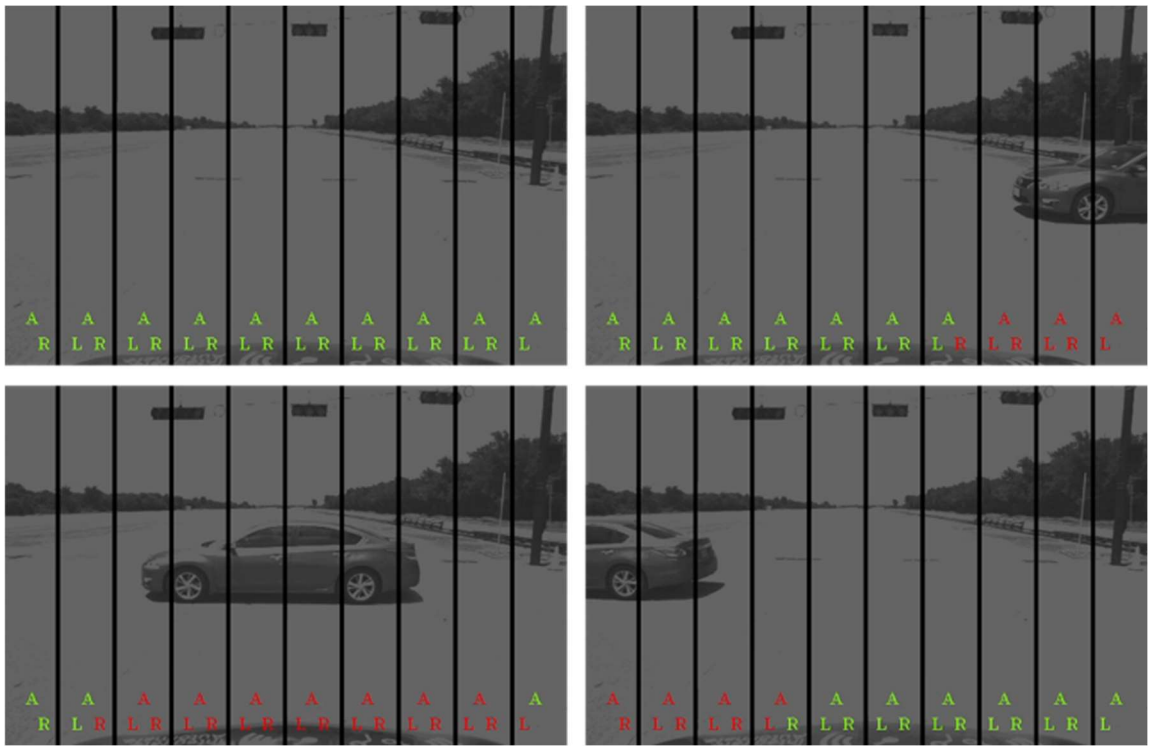


Figure 2.7 Experiment results: Moving object detected by correlation coefficient.

Since controllers can be provided with information on the size and location of objects moving along with risk levels, it will be easier to predict and prepare for future situations.

Figure 2.7 shows the results of recognizing the movement of objects from each section itself or from adjacent sections by calculating the correlation coefficient in the case of Scenario 1. "A" displayed at the bottom of each figure is auto-correlation, which represents motion detection inside the section, and "L" and "R" represent motion change detection from the left and right frames, respectively. As can be seen in Figure 2.7, when a vehicle passes from right to left, when movement is detected, the markings "A", "L", or "R" change to red.

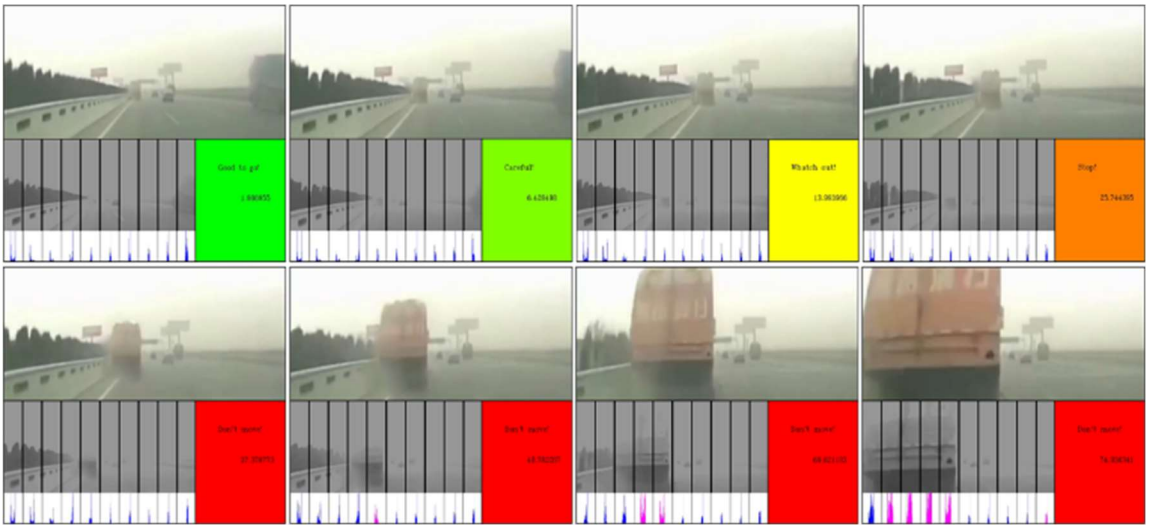


Figure 2.8 Experiment results: Tesla crash dashboard camera footage.

Note that the experiment was focused on the situations when the vehicle is not moving, only the other vehicles are moving. This was done to establish the baseline performance of the proposed EWS. One can extend the applications of the system to the controlled vehicle that moves but certain adjustments need to be made including enhanced segmentation to move away from purely vertical segments to parallelograms that will fit the angle of view and the road horizon. For this adjustment adding GPS data and maps will be needed to secure effective image segmentation.

Figure 2.8 shows the results of the additional experiment on the dashboard camera video of the accident vehicle of Tesla Model Sedan in China in January 2016. It seems clear that the method proposed in this paper successfully confirmed that the warning signal occurred, and through this, the accident could be prevented. Of course, although it is not yet confirmed whether Tesla's Autopilot function operates when a corresponding accident occurs, it seems certain that no warning message has been posted

to the driver, so the supplementary early warning system proposed by this paper can be said to be an essential double safety device.

Many existing self-driving cars and their controllers, including Tesla Autopilot, claim that they rely solely on object detection in machine vision and that their performance is sufficient, but as explained in the previous section of this paper, there are way too many cases when object detection fails. The proposed in this paper, shows promising results and is unlikely to fail in recognition of moving. Of course, pixel-based systems may be sensitive to solar illumination and weather conditions, but camera sensors that have been much improved recently. Therefore, it is suggested that if the pixel-based risk warning system introduced in this paper is used simultaneously with the use of basic object detection, the probability of failure of object detection can be greatly reduced, and a more reliable self-driving system can be developed.

2.6. Summary and recommendations

This paper presents a method that can be used as an early warning system for detecting objects entering vehicle's collision path. As such it is particularly suitable for use in autonomous i.e., self-driving cars. The overall method employs only images as data and relies on the analysis of the time series behavior of pixels within defined segments. Hence, as the proposed method is only based on pixel behavior and can be easily done in real time, it represents a robust supplement to the existing collision avoidance systems that are based on machine learning methods and R-CNN.

The proposed method is also only a first step in developing pixel/segments based early warning systems. In this step we established the baseline performance of the system without changing too many variables. The field experiments done on TAMU testing ground shows promising results. The vehicles can be easily detected without the use of data intensive machine learning methods.

In future we plan to expend the experiment to include two important extensions: 1. the use of a camera with a wider view, such as a fish-eye camera, and 2. conduct experiments with different configurations of driving environment, both geometry and background.

2.7. References

[1] B. Vlasic and N. E. Boudette, "Self-driving Tesla was involved in fatal crash, U.S. says," The New York Times, June 30, 2016. [Online]. Available: https://www.nytimes.com/2016/07/01/business/self-driving-tesla-fatal-crash-investigation.html?module=inline, Accessed on: Sep. 5, 2021.

[2] E. Pettersson and D. Hull, "Tesla sued over fatal crash blamed on Autopilot malfunction," Bloomberg, May 1, 2019. [Online]. Available: https://www.bloomberg.com/news/articles/2019-05-01/tesla-sued-over-fatal-crash-blamed-on-autopilot-navigation-error, Accessed on: Sep. 5, 2021.

[3] "Tesla hit parked police car 'while using Autopilot'," BBC, May 30, 2018. [Online]. Available: https://www.bbc.com/news/technology-44300952, Accessed on: Sep. 5, 2021.

[4] R. Geirhos et al., “Comparing deep neural networks against humans: object recognition when the signal gets weaker,” arXiv:1706.06969, 2017.

[5] R. Girshick et al., "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580-587, DOI: 10.1109/CVPR.2014.81.

[6] R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440-1448, DOI: 10.1109/ICCV.2015.169.

[7] S. Ren et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, DOI: 10.1109/TPAMI.2016.2577031.

[8] S. Lefèvre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” ROBOMECH Journal, vol. 1, no. 1, 2014.

[9] C.-F. Lin, A. G. Ulsoy and D. J. LeBlanc, "Vehicle dynamics and external disturbance estimation for vehicle path prediction," in IEEE Transactions on Control Systems Technology, vol. 8, no. 3, pp. 508-518, May 2000, DOI: 10.1109/87.845881.

[10] R. Schubert, E. Richter and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," 2008 11th International Conference on Information Fusion, 2008, pp. 1-6.

[11] K. Driggs-Campbell and R. Bajcsy, "Identifying Modes of Intent from Driver Behaviors in Dynamic Environments," 2015 IEEE 18th International Conference

on Intelligent Transportation Systems, 2015, pp. 739-744, DOI: 10.1109/ITSC.2015.125.

[12] S. B. Amsalu et al., "Driver behavior modeling near intersections using support vector machines based on statistical feature extraction," 2015 IEEE Intelligent Vehicles Symposium (IV), 2015, pp. 1270-1275, DOI: 10.1109/IVS.2015.7225857.

[13] R. Regele, "Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles," Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08), 2008, pp. 94-99, DOI: 10.1109/ICAS.2008.10.

[14] M. Hülsen, J. M. Zöllner and C. Weiss, "Traffic intersection situation description ontology for advanced driver assistance," 2011 IEEE Intelligent Vehicles Symposium (IV), 2011, pp. 993-999, DOI: 10.1109/IVS.2011.5940415.

[15] G. Wang, D. Xiao and J. Gu, "Review on vehicle detection based on video for traffic surveillance," 2008 IEEE International Conference on Automation and Logistics, 2008, pp. 2961-2966, DOI: 10.1109/ICAL.2008.4636684.

[16] C. R. Wren et al., "Pfinder: real-time tracking of the human body," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 780-785, July 1997, DOI: 10.1109/34.598236.

[17] A. Elgammal et al., "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," in Proceedings of the IEEE, vol. 90, no. 7, pp. 1151-1163, July 2002, DOI: 10.1109/JPROC.2002.801448.

[18] M. Heikkila and M. Pietikainen, "A texture-based method for modeling the background and detecting moving objects," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 4, pp. 657-662, April 2006, DOI: 10.1109/TPAMI.2006.68.

[19] G. Jäger and M. Füllsack, “Systematically false positives in early warning signal analysis,” PLOS ONE, vol. 14, no. 2, 2019.

[20] M. Scheffer et al., “Early-warning signals for critical transitions,” Nature, vol. 461, no. 7260, pp. 53–59, 2009.

[21] R. C. Eisenmann, Machinery malfunction diagnosis and correction: Vibration analysis and troubleshooting for the process industries. Estados Unidos: Hewlett-Packard Company, 1998.

[22] W. Blaschke et al., Stress testing of financial systems: an overview of issues, methodologies, and FSAP experiences. Washington, DC: IMF, 2001.

[23] N. H. Fletcher, “Autonomous vibration of simple pressure-controlled valves in gas flows,” The Journal of the Acoustical Society of America, vol. 93, no. 4, pp. 2172–2180, 1993.

[24] I. Damnjanovic and T. Aven, “Critical slowing-down framework for monitoring early warning signs of surprise and unforeseen events,” Knowledge in Risk Assessment and Management, pp. 81–101, 2017.

[25] R. Felton, “Tow years on, a father is still fighting Tesla over Autopilot and his son’s fatal crash,” Jalopnik, Feb. 27, 2018. [Online]. Available:

https://jalopnik.com/two-years-on-a-father-is-still-fighting-tesla-over-aut-1823189786, Accessed on: Sep. 8, 2021.

3. STEERING GEARS CONFIGURATIONS IN PURE PURSUIT METHOD FOR AUTONOMOUS GROUND VEHICLES

3.1. Introduction

Advances in sensing technologies, computing, and communications systems are making autonomous ground vehicles i.e., self-driving cars increasingly viable technology to solve some of the great challenges posed by modern transportation. [1] The autonomous and connected vehicles are currently being developed and tested for improved safety, reduced carbon emission, and efficient traffic flow among many other important outcomes. [2][3] It is a general expectation that in near future autonomous and connected systems will be become mainstream vehicle technology for the majority of automotive OEMs. [4]

Complex and highly segmented market ecosystems follow the development of autonomous and connected systems. [5] Cadillac's Super Cruise is one of the most visible and media-present companies promoting the self-driving features of its vehicles. In fact, Cadillac's Super Cruise is credited to be the key differentiator making customers choose Super Cruise over other OEMs. [6] However, other major OEMs including Tesla, Ford, Daimler Benz, and Volvo are not too behind as they are also heavily investing in self-driving systems. In fact, not only automotive OEMs play important role in self-driving systems but also many of the big-tech companies such as Apple, Uber, and Google Waymo compete in developing self-driving cars and building business models around the concept. The autonomous and connected vehicle market ecosystem is broad

and deep to include many of the key players in the semiconductor, software, sensing industries.

Despite technological progress and the elaborate market ecosystem, the promise of the self-driving revolution has yet to materialize. [7] In fact, the complexities steaming from often unpredictable driving environment makes the implementation of the developed algorithms challenging. For example, the driving environment is an open system subjected to many interfaces such as weather and wildlife, roadway geometry is often inconsistent, while the traffic behavior is based on human perception and reaction that is defined by bounded rationality. This makes the AI algorithms less effective when compared to their performance in closed system games such as chess, checkers, and the Chinese game of Go. Furthermore, highly publicized accidents in which autonomous vehicles have failed to turn in sharp curves, or recognize pedestrians, median barriers, or even large 18-wheeler trucks have started to undermine public acceptance of the technology. To address these technical challenges, one first needs to fully understand the system architecture and its components.

By definition, autonomous vehicles are mission-oriented self-actuating vehicles capable of perceiving and reacting to the environment without human assistance. In the process of self-navigation, the system relies in general on four sequential activities: perception, localization, path planning, and control. These components are fundamental and translated from the field of robotics. The machine i.e., the vehicle first perceives the environment including roadway geometry, vehicles, and other static and/or dynamic objects, then it locates itself within this environment often with the help of HD maps.

Once the location and the constraints are defined, the path of the machine/vehicle is determined i.e., planned. The plan is then executed with a predefined control strategy.

In the Control phase, the vehicle will begin full-scale operation by determining engine running and direction of travel based on the prepared planned path. If perception is said to be sensory organs such as eyes and ears, control is the brain that determines movement and gives orders. There are large categories of steering and acceleration techniques for controlling the vehicle's movement, which are techniques for steering the steering angle, and acceleration or deceleration/stopping the vehicle through acceleration and braking.

Despite extraordinary effort by the professionals and researchers the development of vehicles' controllers has still ways to go as ground vehicles unlike drones are subjected to the roadway conditions that can be highly unpredictable and inconsistent. For example, not all roads, especially rural roads, are designed with proper transition curves and curvatures for the prevalent speeds. With a wrong speed assumption and lack of transition curves, even highly complex controllers will struggle to keep the vehicle within lane boundaries.

Unlike in robotics, where the feasible region for path planning is determined and rarely updated, in roadway vehicles, the feasible region is only informative of future turning strategies and will likely change based on new information from driving environment i.e., curvature, traffic, pavement conditions. Furthermore, the future speed of the roadway vehicle depends on the future geometry as the roadway design elements are determined to accommodate the turning maneuvers at design speeds. In other words,

the perceived roadway geometry is telling us when and how to turn. This is quite different than how drones and other robots are controlled.

Some self-driving systems [8][9] provide path planning phase in the form of transition curves using Clothoid or Euler function to implement similar driving techniques. However, in this case, the controller does not know the characteristics of the actual road directly and relies solely on the path passed by the path planner, and if the route planning fails or the wrong path is taken over, the overall vehicle operation fails. Therefore, this paper seeks to address the controller role when a given path is established from the path planning stage. To this end, we would like to look at the advantages and disadvantages of the typical controllers used in various self-driving car industries and introduce new ways to take advantage of the Pure Pursuit Controller that we want to use in this paper, and to make up for the vulnerability with the newly designed Automatic Gear Selection for Pure Pursuit method.

3.2. Background

3.2.1. Path following controllers

There are various methods of controlling vehicle steering, such as the Pure Pursuit, Stanley Method, Dynamic Control, Linear Quadratic Regulator (LQR) Control, and Optimal Preview Control. Pure Pursuit and Stanley method are considered geometrical methods as they use the geometric relationship between the vehicle and the path to follow the path and uses the current vehicle position, as well as the azimuth to

select a suitable lookahead point on a given path and calculate the steering angle to follow it.

The Pure Pursuit, first discussed in [10], is a method of using path tracking, which involves geometrically calculating the curvature of the arc between the position of the rear axle of the vehicle and the lookahead point to be carried out by the vehicle. The coordinates of the lookahead point are determined from the position of the rear axle a distance away from the position of the axle in the direction the vehicle wishes to proceed. Once the lookahead point is established, the steering angle of the vehicle is determined by the angular difference between the vector in the direction the vehicle is in and the vector in the direction towards the lookahead point. Pure Pursuit is very useful for large errors and discontinuous route tracking. [11] However, the question of how to select a lookahead point remains a big question. Most self-driving systems using Pure Pursuit use the vehicle's current speed to determine this lookahead point which creates the problem of cutting corners and overshooting.

The Stanley method is the route tracking method used in unmanned vehicles developed by Stanford University to target the DARPA Grand Challenge. [12] This method uses a nonlinear feedback function for route departure errors that can be measured from the center of the front axle to the point on the nearest driving path. The Stanley method has the simplest yet best performance. Although it has better performance than Pure Pursuit in many ways, the Stanley method has the disadvantage of being vulnerable to large errors and discrete paths. [11]

Kinematic Controller using the Kinematic cycle model, and Optimal Preview Control and Linear Quadratic Regulator (LQR) using the Dynamic cycle model are methods for calculating control values using the kinetic properties of each vehicle model. The Kinematic controller has the advantage of not doing cutting corners, a problem with Pure Pursuit, but it has a weak disadvantage with robustness to disturbances. [11] The LQR method and the Optimal method have the advantage of being easy to understand and implement, but because they have the limitations of the Dynamic cycle model, there are problems with poor performance at low speed and overshooting of sudden changes in curvature. [11]

Table 3.1 shows the pros and cons of the controllers mentioned above. Note here that, unlike other controllers, Pure Pursuit has excellent results in the remaining criteria except cutting corners and overshooting, which are items of driving error in the nature of its algorithm. In other words, the Pure Pursuit method would be much better than other controllers, which would have prompted two vehicles to use Pure Pursuit at DARPA Grand Challenge [13] and three vehicles at DARPA Urban Challenge. [14][15]

Table 3.1 Path-following Algorithms Pros and Cons

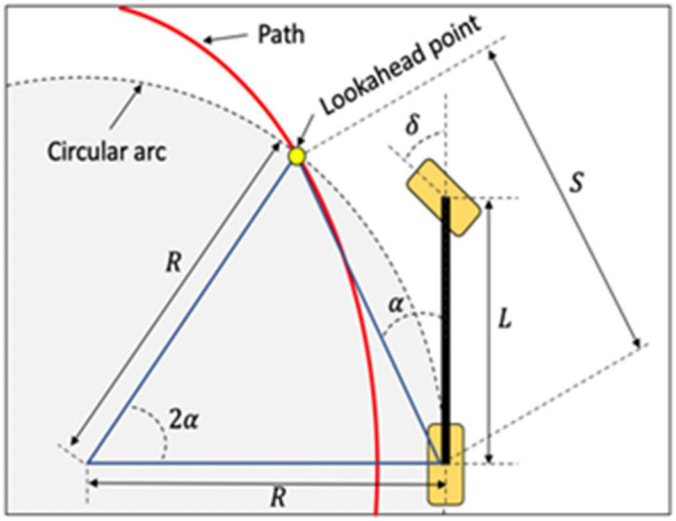
	Pros	**Cons**
Pure Pursuit	- Works fairly well. - Quite robust to large errors and discontinuous paths.	- Not clear how to pick the best look-ahead distance.
Stanley	- Simplest - Performs surprisingly well. (Outperforms Pure Pursuit in most scenarios)	- Not as robust to large errors and non-smooth paths.
Kinematic	- Easily extended for applications that need to pull a path following the trailer.	- A Little more difficult to understand and implement than the Pure Pursuit or Stanley method.

Table 3.1 Continued

	Pros	Cons
Kinematic (Continued)	- Can be applied to a large class of mobile robots approximated by kinematic models written in the chained form. - For the car-like robot application, the overall tracking performance and robustness at moderate speed is comparable to the Stanley method	- Significant increases in the online computational works needed to compute the steering angle velocity.
MPC	- Plant dynamics can be fully exploited - Generic consideration of complex control goals - Simple control policy for complex systems - Generic consideration of constraints	- A plant model is required - High computational load - high algorithmic complexity - High number of control parameters
LQR with Feed-Forward	- Easy to understand and implement. - Great choice for highway driving and many urban driving scenarios.	- Not considerably outperform the Stanley or Kinematic methods in many scenarios. - Not perform as well at very low speeds. - Significant overshoot occurs during rapid, even smooth, changes in path curvature. - Not robust like the Pure pursuit or Stanley method to large errors or path discontinuity.
Optimal Preview	- Little more complicated than the LQR method, but it is still easy to understand and implement. - Provides the LQR method with a look-ahead, or preview, of the upcoming path to address the problem.	- Constant velocity assumption.

3.2.2. Pure pursuit controller

Pure Pursuit is the most common and frequently used controller in the path tracking method of mobile robots. In fact, it is the most widely used method for challenging environments including Roborace and Indy Autonomous Challenge. It is especially useful when a High-Definition digital map is not available and unusable due to the factors such as high speed. As described earlier, Pure Pursuit is a method that



where

S : Lookahead Distance

L : Vehicle Length

R : Radius of the Curve

ρ : Curvature of the Curve (ρ=1/ R)

α : Angle between Vehicle Heading and direction of Lookahead Point

Figure 3.1 Basic Pure pursuit set up with using Bicycle Model

utilizes the curvature of the circle through which the lookahead point on the front path and the rear axle of the vehicle passes. The curvature of the circle is determined by the lookahead distance S. (Figure 3.1) Procedure for obtaining steering angle (δ) using Pure Pursuit is as follows.

$$\frac{S}{\sin(2\alpha)} = \frac{R}{\sin\left(\frac{\pi}{2}-\alpha\right)} \tag{3.1}$$

$$\frac{S}{2\sin(\alpha)\cos(\alpha)} = \frac{R}{\cos\alpha} \tag{3.2}$$

$$\frac{S}{\sin(\alpha)} = 2R \tag{3.3}$$

$$R = \frac{S}{2\,\mathrm{si}\ \ (\alpha)} \tag{3.4}$$

$$\rho = \frac{1}{R} = \frac{2\sin(\alpha)}{S} \tag{3.5}$$

By applying the curvature ρ to the geometric bicycle model of a vehicle with Ackermann's steering angle we have:

$$\delta = \tan^{-1}(\rho L) \tag{3.6}$$

$$\delta = \tan^{-1}\left(\frac{2L\sin(\alpha)}{S}\right) \tag{3.7}$$

The result δ is finally the steering angle for the vehicle to follow its path.

As it can be observed from Eq. 7 lookahead-distance, S, plays the most important role in Pure Pursuit. The long or short S changes the following target point, and the curvature of the circle passing the following target point and the rear-wheel axis. If the S is long, the following target point is selected far away and the circular curvature across the following target point and rear-wheel axis increases, making the steering angle smooth and the steering angle less shaky, thus stabilizing the vehicle's behavior. However, if S is excessively long, path-following performance is degraded. In the opposite case, a shorter S means that the following target point is close, thus reducing the circle curvature across the following target point and rear-wheel axis, allowing quick access to the path and improving path follow performance. But again, the shorter the S, the more severe the steering angle changes, and in excessive cases, lateral control is emitted, causing the vehicle to behave in a dangerous manner. As the question of whether to hold S short or long is directly related to the trade-off problem of stability and tracking performance, it is very important to choose the optimal S.

3.2.3. Transition Curve

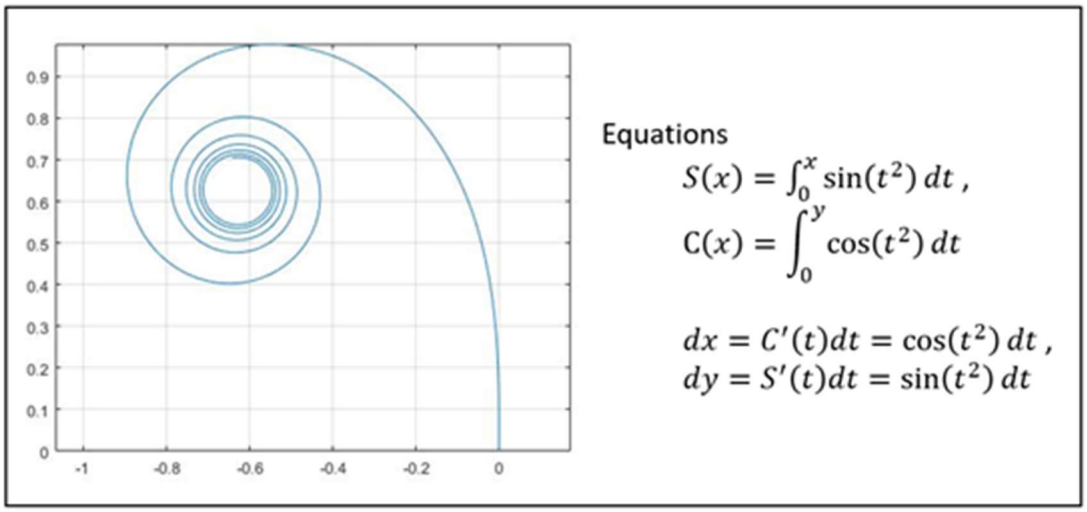


Figure 3.2 Clothoid curve

Transition curve is as known as Clothoid, Cornu spiral, or Fresnel integrals. The introduction of gentle curves to prevent accidents caused by sudden curves when a straight road meets a curved road with a specific radius. When designing all vehicle roads, the transition curve is used to mitigate drastic curvature changes.

In the case of a vehicle, even in the case of a non-transition curve, the path to the curve entry of all vehicles will take the form of the transition curve. Transition curve is essential, especially for rail designs on trains with no degree of freedom over the route. Figure 3.2 shows the form of a Clothoid curve and the formula that constitutes it.

3.3. Methodology

As mentioned before, Pure Pursuit may or may not show a good path-following performance depending on the lookahead-distance, choice of speed, and the type of geometry. In each case, there is a trade-off, which means that choosing a good path-

following performance increases the lateral-jerk, while minimizing the lateral-jerk decreases the path-following performance. We would like to propose two different ways to compensate for the shortcomings of the original Pure Pursuit: 1. Steering Gear Pure Pursuit and 2. Automatic Gear Selection Pure Pursuit.

3.3.1. Steering Gear Pure pursuit

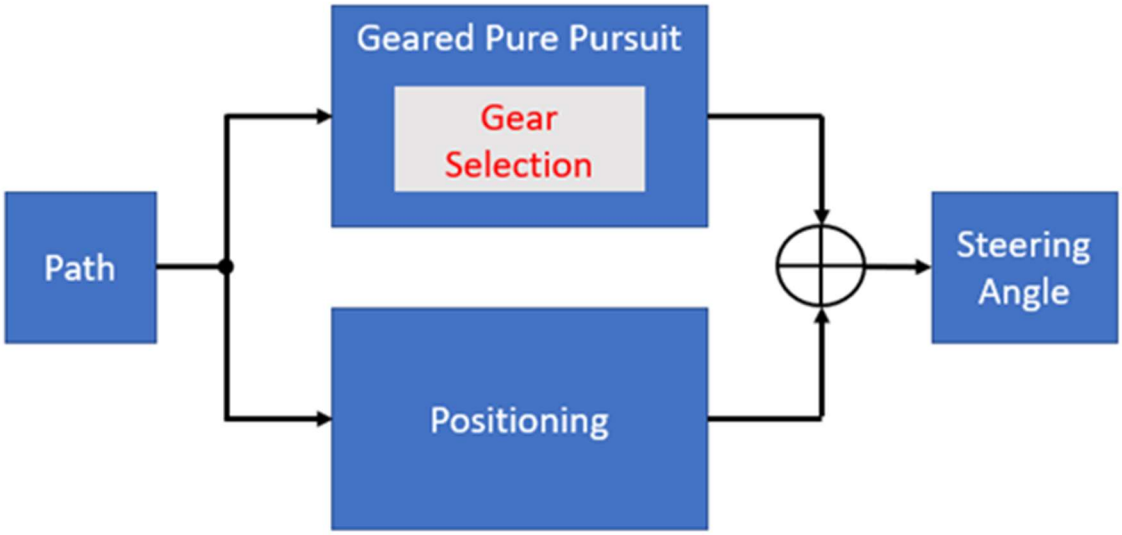


Figure 3.3 Steering Gear Pure pursuit

Figure 3.3 shows the first proposed method of a novel path-following controller. Receiving the paths provided in the cognitive and path planning phases, two different methods of computation are applied, in which the final steering angle is derived by summing the steering angles calculated.

3.3.1.1. Calculate steering angle to follow roadway geometry

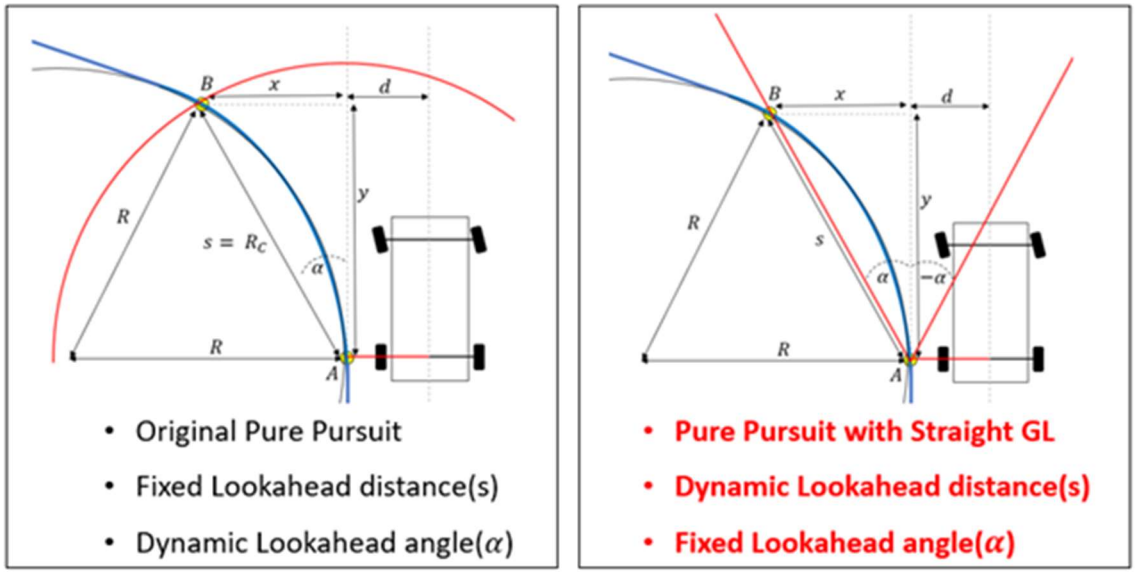


Figure 3.4 Pure pursuit using Circular-guideline(left), and Straight-guideline(right)

As shown on the left side of Figure 3.4, conventional Pure Pursuit uses a Circular-guideline that uses a fixed radius to find the lookahead point. Because of its fixed radius, the lookahead-distance is used as a fixed value, but the lookahead-angle is a variable. The problem, in this case, is that if the lookahead-angle, which has a greater effect on the change in steering angle, changes rapidly, the calculated final steering angle will also change rapidly, resulting in a larger lateral-jerk.

To determine the variable lookahead-distance, we use a new method using Straight-guidelines as shown on the right side of Figure 3.4. Because the Straight-guideline uses a straight line with fixed angles, it can fix the lookahead-angle and use variable lookahead-distance at the same time. Of course, the use of the Straight-guideline is not perfect. Because the lookahead-angle is fixed, the lookahead point cannot be specified unless the angle and path cross. We address this only disadvantage not by using a single angle of Straight-guideline, but by using multiple angles of

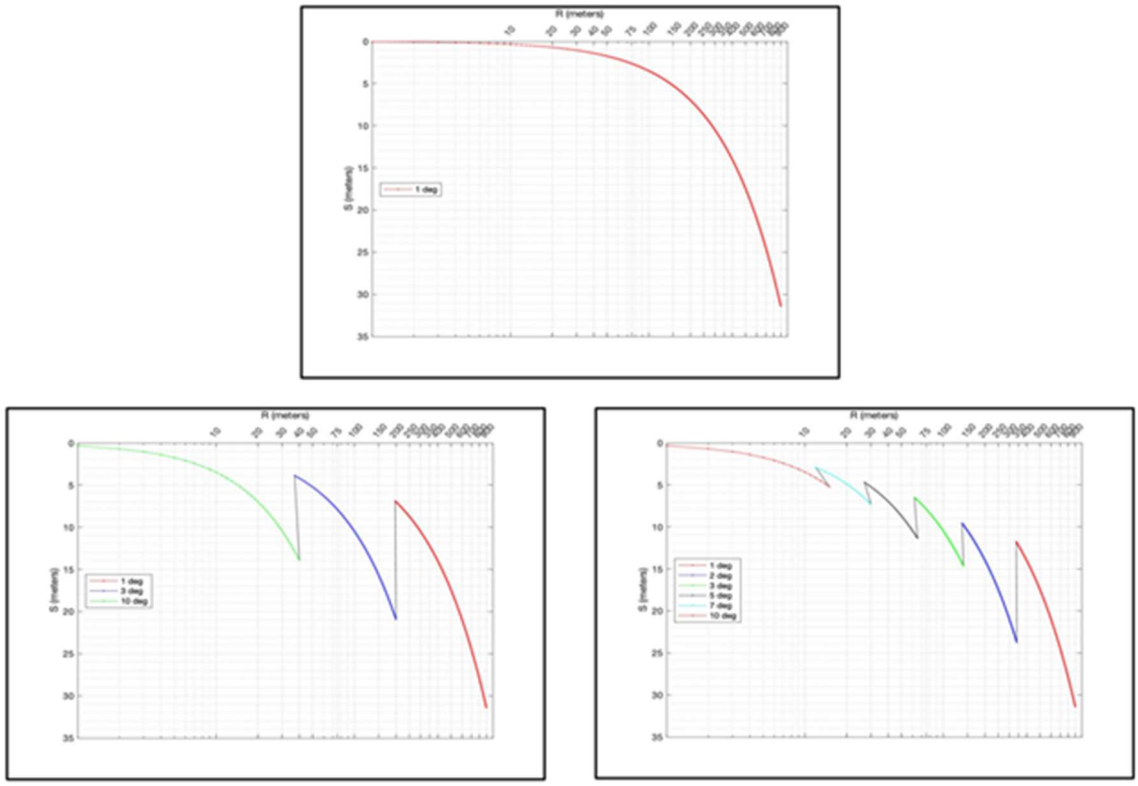


Figure 3.5 Three different gear configurations. Single gear(top), three gears(bottom left), and six gears(bottom right)

Straight-guideline. Three types of combinations of Straight-guidelines were prepared for the experiment: One Straight-guideline with an angle of 1°, 6 Straight-guidelines with angles of 1°, 3°, and 10°, and 6 Straight-guidelines with angles of 1°, 2°, 3°, 5°, 7°, and 10°.

So, what is 'Gear' we are talking about? We named 'Gear' the Straight-guideline with each different lookahead-angle and produced a new way of changing gears to fit the curvature of the following path, like the gear shift of the vehicle transmission. Figure 3.5 shows Single gear, three gears, and six gears configurations. We designed to use different gears depending on the radius of the curve being detected, as shown in Figure

alpha (deg)	R (meters)	10	20	40	80	160	320	640
1	S(R)	0.34904813	0.69809626	1.39619251	2.79238503	5.58477006	11.1695401	22.3390802
	S(R)-10cm	0.24904813	0.59809626	1.29619251	2.69238503	5.48477006	11.0695401	22.2390802
	S(R)+10cm	0.44904813	0.79809626	1.49619251	2.89238503	5.68477006	11.2695401	22.4390802
	R(S(R)-10cm)	7.13506558	17.1350656	37.1350656	77.1350656	157.135066	317.135066	637.135066
	R(S(R)+10cm)	12.8649344	22.8649344	42.8649344	82.8649344	162.864934	322.864934	642.864934
	R error range	5.72986885	5.72986885	5.72986885	5.72986885	5.72986885	5.72986885	5.72986885
2	S(R)	0.69798993	1.39597987	2.79195974	5.58391947	11.1678389	22.3356779	44.6713558
	S(R)-10cm	0.59798993	1.29597987	2.69195974	5.48391947	11.0678389	22.2356779	44.5713558
	S(R)+10cm	0.79798993	1.49597987	2.89195974	5.68391947	11.2678389	22.4356779	44.7713558
	R(S(R)-10cm)	8.56731458	18.5673146	38.5673146	78.5673146	158.567315	318.567315	638.567315
	R(S(R)+10cm)	11.4326854	21.4326854	41.4326854	81.4326854	161.432685	321.432685	641.432685
	R error range	2.86537083	2.86537083	2.86537083	2.86537083	2.86537083	2.86537083	2.86537083
3	S(R)	1.04671912	2.09343825	4.1868765	8.373753	16.747506	33.495012	66.990024
	S(R)-10cm	0.94671912	1.99343825	4.0868765	8.273753	16.647506	33.395012	66.890024
	S(R)+10cm	1.14671912	2.19343825	4.2868765	8.473753	16.847506	33.595012	67.090024
	R(S(R)-10cm)	9.04463387	19.0446339	39.0446339	79.0446339	159.044634	319.044634	639.044634
	R(S(R)+10cm)	10.9553661	20.9553661	40.9553661	80.9553661	160.955366	320.955366	640.955366
	R error range	1.91073226	1.91073226	1.91073226	1.91073226	1.91073226	1.91073226	1.91073226
5	S(R)	1.74311485	3.48622971	6.97245942	13.9449188	27.8898377	55.7796754	111.559351
	S(R)-10cm	1.64311485	3.38622971	6.87245942	13.8449188	27.7898377	55.6796754	111.459351
	S(R)+10cm	1.84311485	3.58622971	7.07245942	14.0449188	27.9898377	55.8796754	111.659351
	R(S(R)-10cm)	9.42631434	19.4263143	39.4263143	79.4263143	159.426314	319.426314	639.426314
	R(S(R)+10cm)	10.5736857	20.5736857	40.5736857	80.5736857	160.573686	320.573686	640.573686
	R error range	1.14737132	1.14737132	1.14737132	1.14737132	1.14737132	1.14737132	1.14737132
7	S(R)	2.43738687	4.87477374	9.74954747	19.4990949	38.9981899	77.9963798	155.99276
	S(R)-10cm	2.33738687	4.77477374	9.64954747	19.3990949	38.8981899	77.8963798	155.89276
	S(R)+10cm	2.53738687	4.97477374	9.84954747	19.5990949	39.0981899	78.0963798	156.09276
	R(S(R)-10cm)	9.58972455	19.5897245	39.5897245	79.5897245	159.589725	319.589725	639.589725
	R(S(R)+10cm)	10.4102755	20.4102755	40.4102755	80.4102755	160.410275	320.410275	640.410275
	R error range	0.8205509	0.8205509	0.8205509	0.8205509	0.8205509	0.8205509	0.8205509
10	S(R)	3.47296355	6.94592711	13.8918542	27.7837084	55.5674169	111.134834	222.269667
	S(R)-10cm	3.37296355	6.84592711	13.7918542	27.6837084	55.4674169	111.034834	222.169667
	S(R)+10cm	3.57296355	7.04592711	13.9918542	27.8837084	55.6674169	111.234834	222.369667
	R(S(R)-10cm)	9.71206148	19.7120615	39.7120615	79.7120615	159.712061	319.712061	639.712061
	R(S(R)+10cm)	10.2879385	20.2879385	40.2879385	80.2879385	160.287939	320.287939	640.287939
	R error range	0.57587705	0.57587705	0.57587705	0.57587705	0.57587705	0.57587705	0.57587705

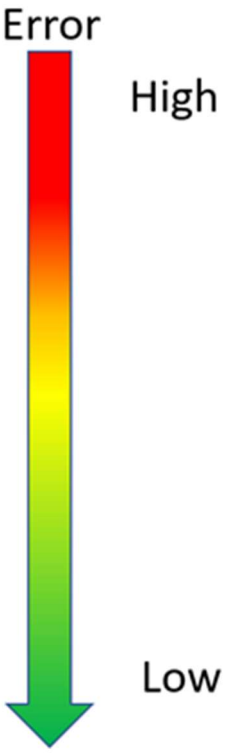


Figure 3.6 Curvature reading error range for each gears

3.5, before solving the problem of which gear to choose when using more than one gear. This is based on the use of a Straight-guideline using small angles when the curvature of the curve is large and is designed to use a Straight-guideline with larger angles as the curve is milder. The smaller the curve curvature, i.e., the closer it gets to a straight road, the smaller the left or right bias when the path becomes similar to the straight-line shape, making it possible to detect the bias of the small curve.

However, the higher angle Straight-guideline allows for more accurate curvature detection. Figure 3.6 shows a result of detecting the curvature of several different curves with different angles of Straight Guideline and measuring the error range of each. As mentioned earlier, the higher the angle of the Straight-guideline, the smaller the error

range. Therefore, we design our Gear Configuration so that we can use a higher angle of Straight-guideline whenever possible.

3.3.1.2. Calculate positioning steering angle

To add more accuracy of path-following ability, one can add the Positioning controller. In addition, this positioning guideline is useful even if all the guidelines we use do not cross paths and thus do not compute the reading of curvature, especially at the end of the non-transition curve. At this moment, the calculated steering angle will be dramatically going to 0 and will cause a dramatic change of the steering angle. The Positioning controller uses the basic Pure Pursuit method and, in addition, uses variable lookahead-distance according to the distance from the car to the path. The radius of the Positioning guideline can be calculated by the following equation.

$$radius_p = 30 * e^{-dist} + dist \tag{3.8}$$

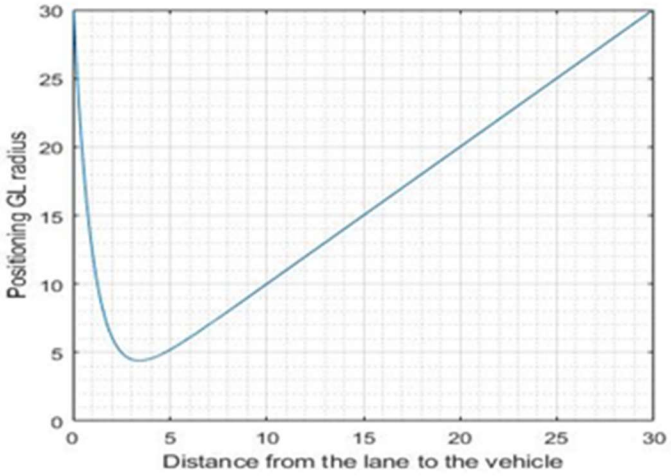


Figure 3.7 Radius of the Positioning guideline

3.3.2. Automatic Gear Selection Pure pursuit

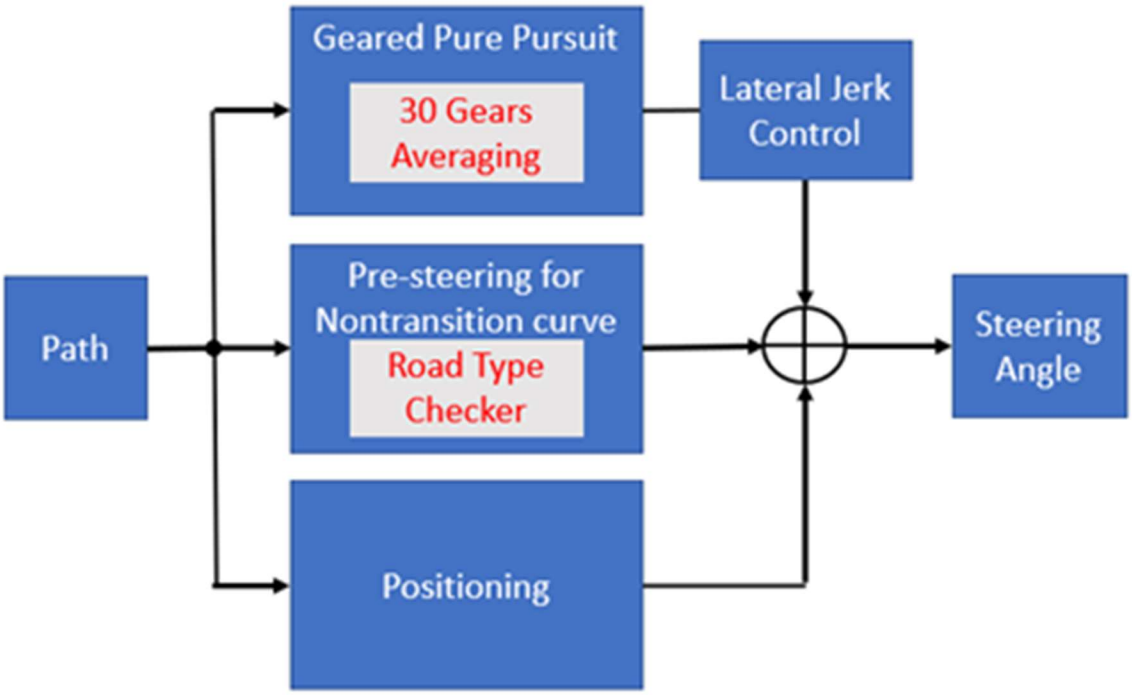


Figure 3.8 Automatic Gear Selection Pure pursuit

Figure 3.8 shows our second proposed method of a novel path-following controller. One big difference from the first novel controller proposed earlier is that instead of using one of the optimal angles of the guideline from up to six different angles, we use all the one-degree increasing from 1 to 30 degrees at once and then calculate their average as the final steering angle. In addition, we determine whether the perceived path is a transition curve, and then use pre-steering only if it is a non-transition curve that causes a large natural curve on entry and escape, and a lateral-jerk adjustment function which we call HTL (half, then linear). As a final step, we design to enhance the performance of path tracking using Positioning control, which was used in the

previously proposed method. The final derived steering angle is obtained by summing all the steering angles calculated in each step.

3.3.2.1. Calculate steering angle to follow roadway geometry

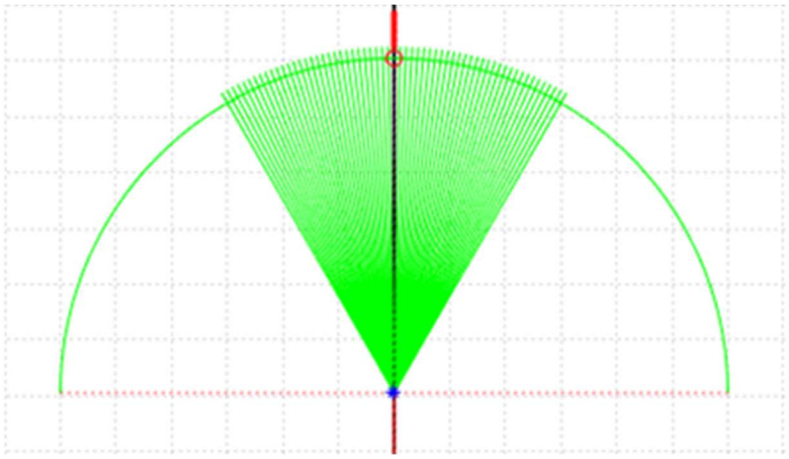


Figure 3.9 Thirty Straight guidelines

Each steering angle is calculated using a total of 30 Straight-guidelines with lookahead-angles ranging from 1 to 30 degrees, and the average of these is obtained to obtain the final steering angle. At this time, we named this method Automatic Gear Selection because it uses all the gears rather than choosing one of the several Straight-guidelines (i.e., Gears).

3.3.2.2. Lateral jerk control

We apply our HTL method so that we can enter and escape curves more smoothly without simply relying on steering angle calculations for fixed geometry following. First, steering the half of difference of curvature from the previous frame until the reading curvature is steady, and then during the same time of half steering, linearly

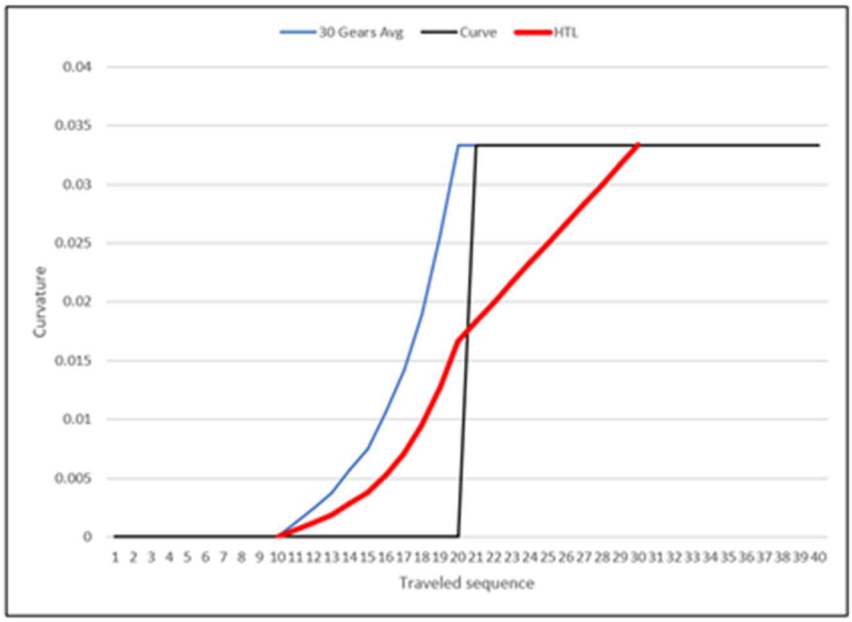


Figure 3.10 HTL (Half, then Linear) method

get to the maximum of its curvature. The red line from Figure 3.10 shows this HTL method and the blue line shows the native curvature reading from the 30 Gears' average.

3.3.2.3. Pre-steering for non-transition curve

In racing competitions and adequate transition curves, it is common to advance to the outside of the lane before entering the curve and then attempt to steer early before the curve begins to avoid oversteering by minimizing the lateral-jerk; the out-in strategy. In addition to racing, we can easily see drivers turning the steering wheel even before the start of the curve when entering a sharp curve. Inspired by this, we added a more advanced approach for steering in the case of non-transition curves where the curve starts sharply to minimize the lateral-jerk. Intuitively following the non-transition curve results in a large lateral-jerk, so to avoid such a large lateral-jerk, the same principle is to

follow a virtual transition curve, which starts steering in advance before the actual curve starts, gradually increasing the steering value to the point where the curve starts.

3.3.2.4. Calculate Positioning steering angle

We improve the accuracy of additional path-following using the same method as the Positioning control method used in the previously proposed Manual Gear Pure Pursuit.

3.4. Experimental setting

For simulation of the proposed methods, we have implemented and experimented with MATLAB. We fixed the steering angular speed and operating frequency of simulation vehicles at 0.4 deg/ms and 0.3sec, respectively, for all methods. The test cases used both a normal speed of 4.5m/s and a fast speed of 9.0m/s, and the simulated path has a 30m radius curve, and we experimented with both with and without the transition curve. In the case of the original Pure Pursuit, we experimented with 5m, 10m, 20m, and 30m lookahead-distance for performance comparison according to the lookahead-distance.

For performance evaluation, we measure and compare the accuracy of path-following and the lateral-jerk. The accuracy of path-following is calculated by the path departure error of path-following, and we set the acceptance criteria to a maximum of 1.0m. This acceptance criterion was derived by Figure 3.11.

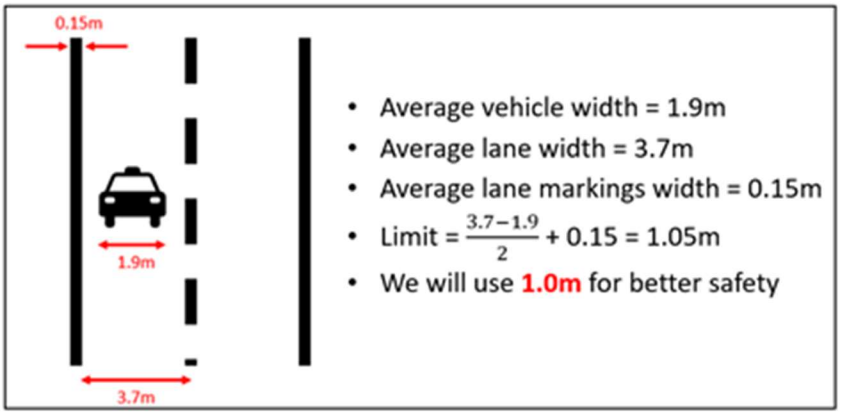


Figure 3.11 Acceptance criteria of path following accuracy

And the lateral-jerk was calculated from the following formula.

$$Lateral\ Jerk = \frac{\Delta acc}{\Delta t} = \frac{(v^2 * \rho)_{curr} - (v^2 * \rho)_{prev}}{\Delta t} \quad (3.10)$$

Generally, acceptable range of non-uncomfortable to the human feel, the lateral-jerk is in the range from 0.8m/sec3 to 1.1m/sec3. We assume 1.0m/sec3 as our limit and confirm the simulation results, considering that we place a greater weight on the safety aspect than only pursuing comfort.

3.5. Results

We measured the distance error and the lateral-jerk as the entire performance gage. 'OK' sign was marked on the figures if the measurement is within the acceptable range (green area), and 'NOK' if it goes into the unacceptable range (red area).

3.5.1. Original Pure pursuit

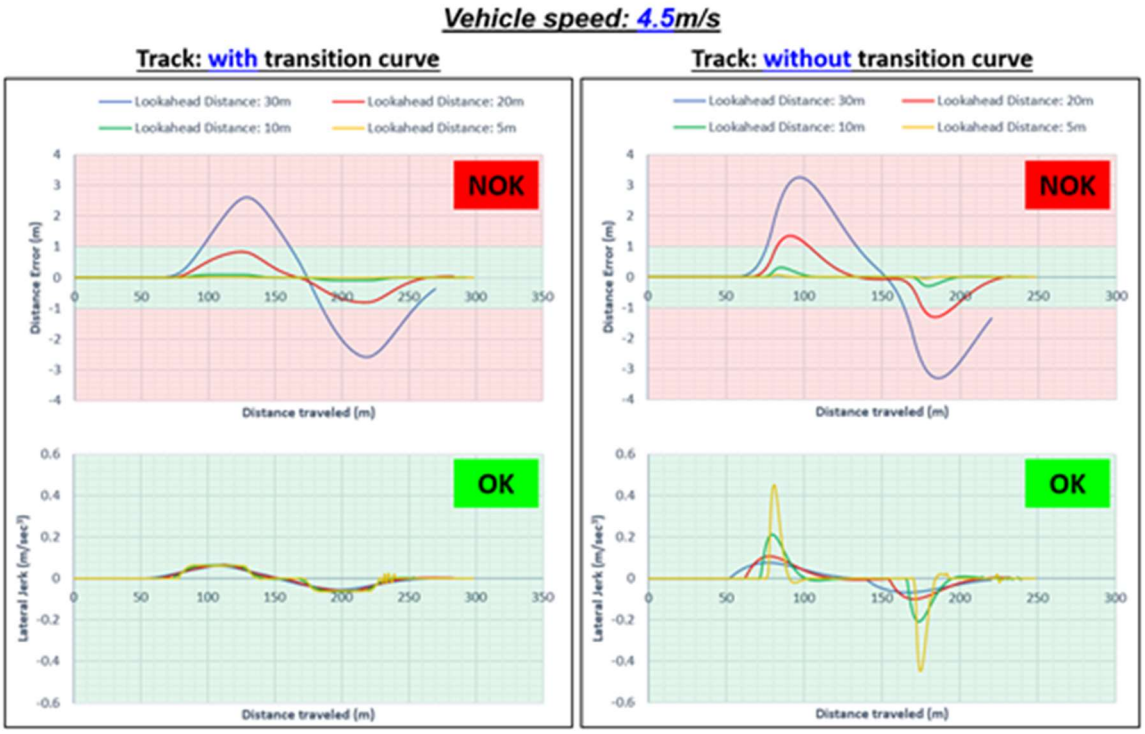


Figure 3.12 Simulation result. Original Pure Pursuit (v=4.5m./s)

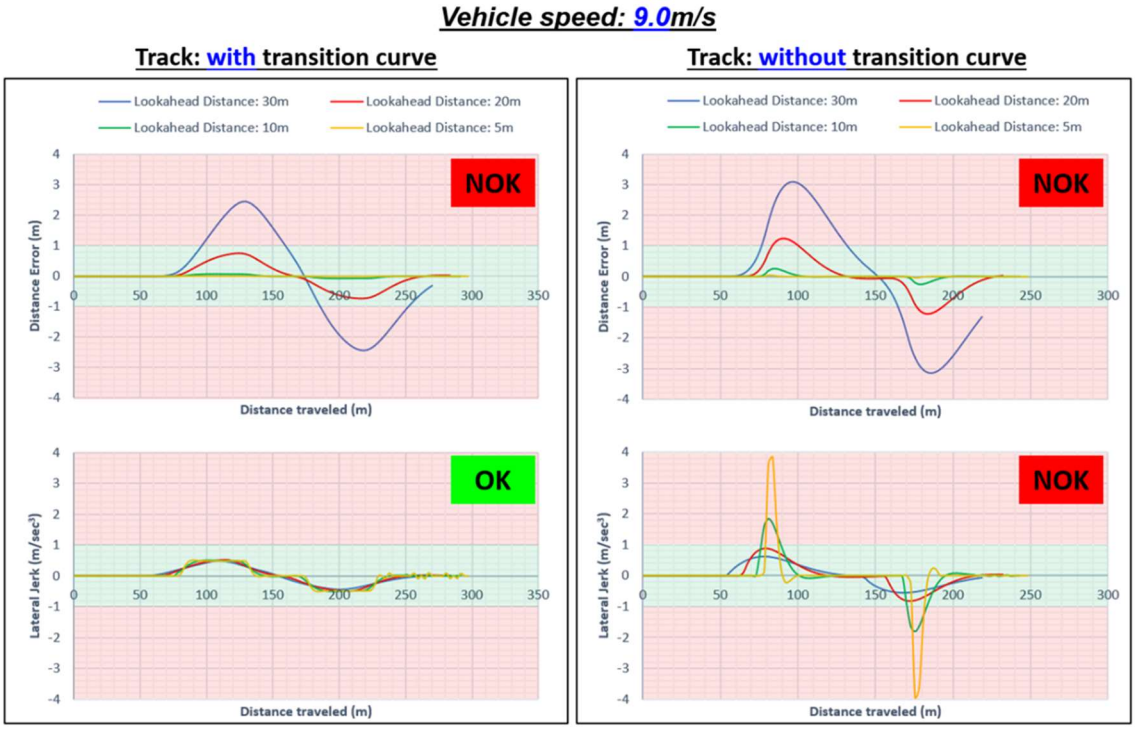


Figure 3.13 Simulation result. Original Pure Pursuit (v=9.0m./s)

The results of the original Pure Pursuit clearly show that there is a trade-off of distance error and lateral-jerk depending on lookahead-distance. (Figure 3.12, Figure 3.13) In the case of the shortest lookahead-distance of 5m, the distance error is very small at all speeds, but in the case of driving at a high speed on a non-transition curve, it is confirmed that a very large lateral-jerk occurs. Conversely, in the case of the longest lookahead-distance of 30m, a distance error occurred significantly out of tolerance regardless of driving speed, but the lateral-jerk had the smallest of the four lookahead-distances we tested.

3.5.2. Steering Gear Pure pursuit

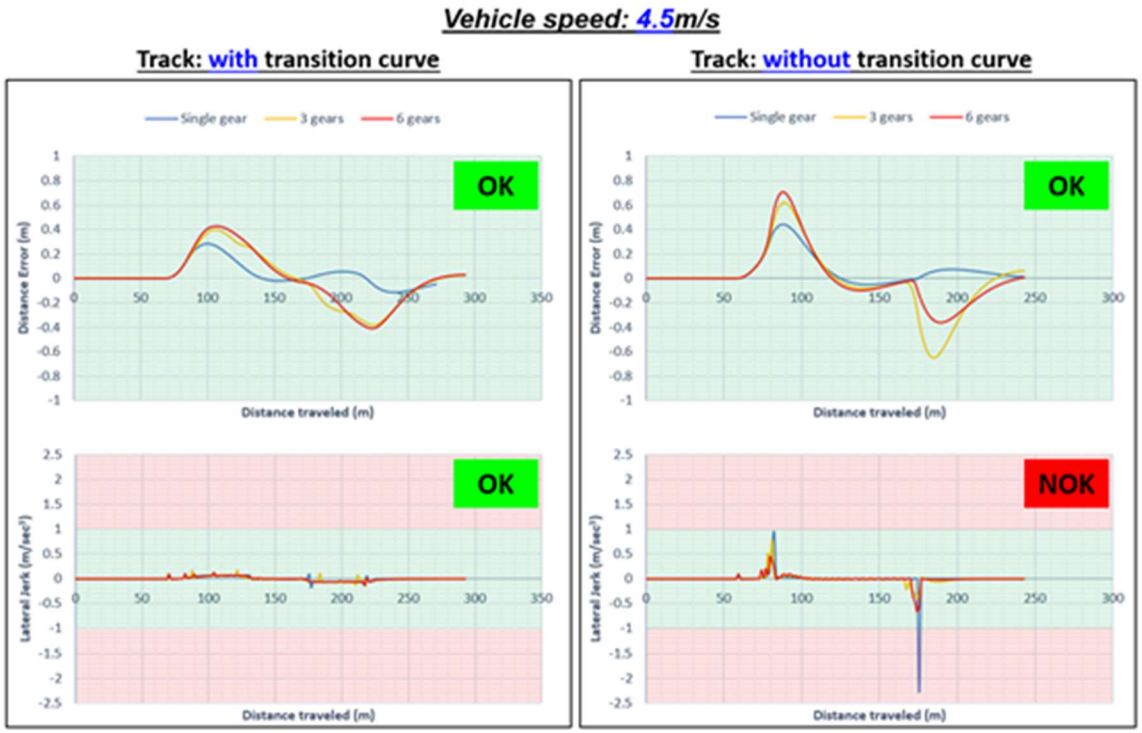


Figure 3.14 Simulation result. Manual Geared Pure Pursuit (v=4.5m./s)

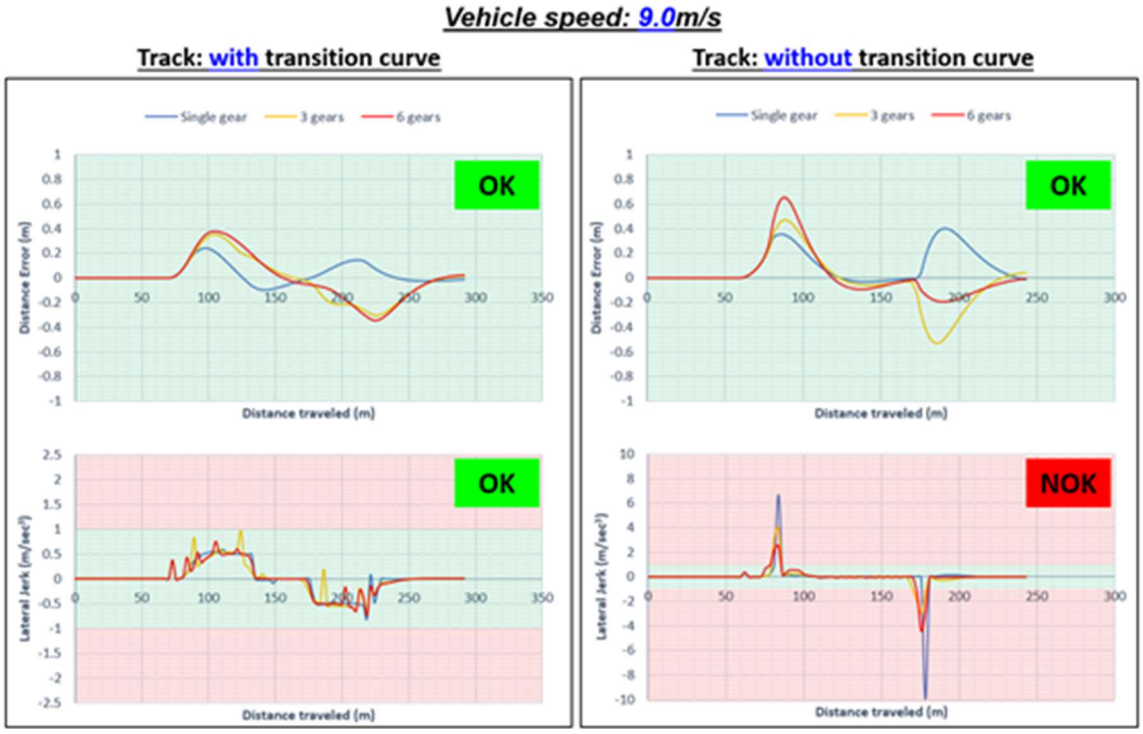


Figure 3.15 Simulation result. Manual Geared Pure Pursuit (v=9.0m./s)

Our first proposed Steering Gear Pure Pursuit showed a slight improvement over the original Pure Pursuit. At slow speeds, both 3 and 6 gears, except for the use of Single gear, came within the tolerance range of distance error and lateral-jerk, but as we expected, single gear showed results that lateral-jerk exceeded the tolerance range on a non-transition curve. (Figure 3.14) However, at high speeds, all three gear configurations resulted in a lateral-jerk greater than the allowable value in a non-transition curve. (Figure 3.15)

3.5.3. Automatic Gear Selection Pure pursuit

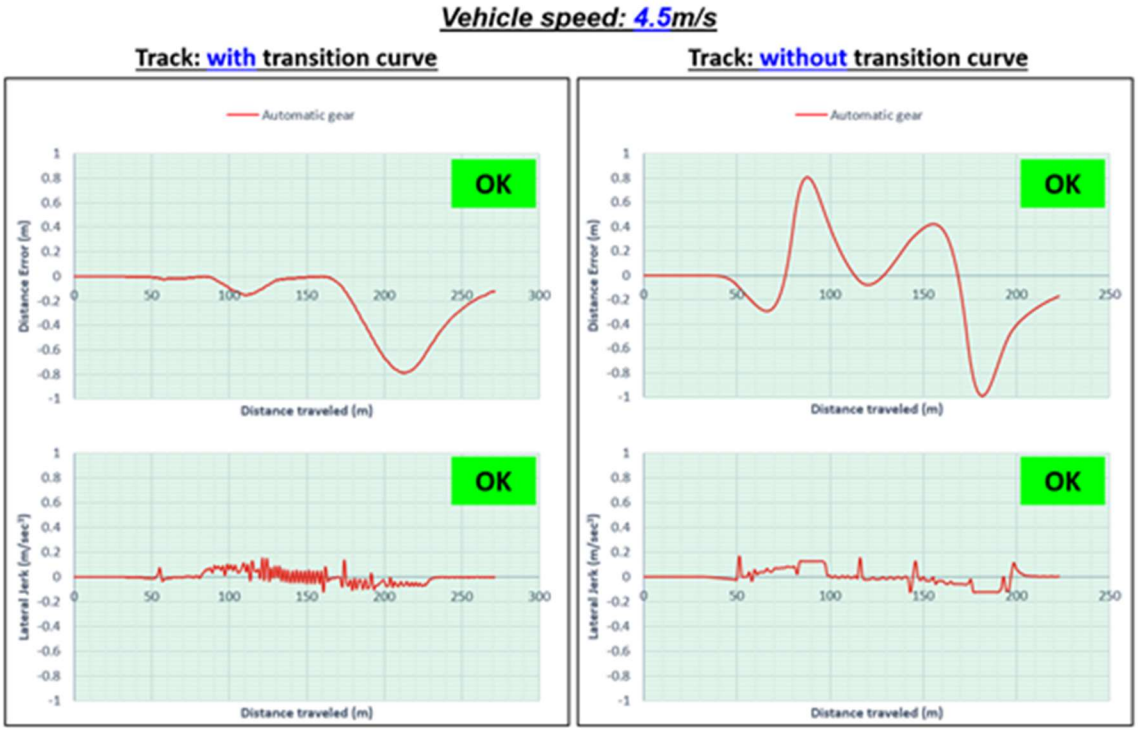


Figure 3.16 Simulation result. Automatic Geared Pure Pursuit (v=4.5m./s)

Figure 3.17 Simulation result. Automatic Geared Pure Pursuit (v=9.0m./s)

Our final proposal, Automatic Gear Selection Pure Pursuit, showed excellent results that both distance error and lateral-jerk did not exceed the allowable range, regardless of the speed and type of curve we experimented with. (Figure 3.16, Figure 3.17)

3.5.4. Overall comparison

		Vehicle Speed = 4.5 m/s (16.2 km/h, 10.0662 mi/h) Wheel Angular Speed = 0.4 deg/ms Operating Frequency = 0.3 sec				Vehicle Speed = 9.0 m/s (32.4 km/h, 20.1324 mi/h) Wheel Angular Speed = 0.4 deg/ms Operating Frequency = 0.3 sec				
	Track	30R Non-Transition Curved		30R Transition Curved		Track	30R Non-Transition Curved		30R Transition Curved	
Algorithm		Departure from the path (m)	Lateral Jerk (m/sec^3)	Departure from the path (m)	Lateral Jerk (m/sec^3)		Departure from the path (m)	Lateral Jerk (m/sec^3)	Departure from the path (m)	Lateral Jerk (m/sec^3)
Original Pure Pursuit Lookahead: 30 (m)	Mean	1.334	0.0302	1.094	0.0237	Mean	1.246	0.2404	1.019	0.1889
	SD	1.246	0.0270	0.960	0.0208	SD	1.187	0.2213	0.905	0.1686
	Max	3.308	0.0762	2.609	0.0604	Max	3.147	0.6206	2.453	0.4856
Original Pure Pursuit Lookahead: 20 (m)	Mean	0.372	0.0283	0.299	0.0226	Mean	0.332	0.2252	0.267	0.1797
	SD	0.463	0.0363	0.310	0.0236	SD	0.427	0.2983	0.281	0.1905
	Max	1.337	0.1067	0.837	0.0655	Max	1.238	0.8816	0.755	0.5237
Original Pure Pursuit Lookahead: 10 (m)	Mean	0.040	0.0274	0.031	0.0212	Mean	0.031	0.2176	0.025	0.1696
	SD	0.083	0.0578	0.039	0.0264	SD	0.068	0.4844	0.031	0.2124
	Max	0.303	0.2127	0.094	0.0636	Max	0.255	1.8404	0.073	0.5083
Original Pure Pursuit Lookahead: 5 (m)	Mean	0.004	0.0268	0.003	0.0210	Mean	0.002	0.2128	0.002	0.1708
	SD	0.012	0.0889	0.004	0.0275	SD	0.008	0.7893	0.002	0.2193
	Max	0.064	0.4511	0.009	0.0634	Max	0.041	3.9232	0.005	0.5088
Manual Geared Pure Pursuit Positioning GL: dynamic (30*exp(-1.5*dist) + dist) — 1 (deg)	Mean	0.074	0.0276	0.059	0.0235	Mean	0.100	0.2317	0.055	0.1710
	SD	0.113	0.1890	0.076	0.0312	SD	0.130	1.2466	0.065	0.2352
	Max	0.442	2.2853	0.283	0.1728	Max	0.401	9.9428	0.241	0.7997
Manual Geared Pure Pursuit Positioning GL: dynamic (30*exp(-1.5*dist) + dist) — 1-3-10 (deg)	Mean	0.156	0.0281	0.139	0.0236	Mean	0.128	0.2106	0.112	0.1711
	SD	0.202	0.0857	0.140	0.0322	SD	0.164	0.6472	0.115	0.2375
	Max	0.650	0.7859	0.397	0.1737	Max	0.529	3.9867	0.352	0.9587
Manual Geared Pure Pursuit Positioning GL: dynamic (30*exp(-1.5*dist) + dist) — 1-2-3-5-7-10 (deg)	Mean	0.138	0.0278	0.140	0.0222	Mean	0.109	0.2104	0.117	0.1683
	SD	0.178	0.0898	0.151	0.0284	SD	0.155	0.6590	0.127	0.2276
	Max	0.706	0.6581	0.426	0.1280	Max	0.650	4.3857	0.376	0.7507
Automatic Geared Pure Pursuit	Mean	0.268	0.0385	0.178	0.0316	Mean	0.280	0.4034	0.177	0.1964
	SD	0.268	0.0455	0.243	0.0354	SD	0.244	0.3756	0.232	0.2368
	Max	0.988	0.1660	0.784	0.1497	Max	0.960	0.9922	0.762	0.9173

Figure 3.18 Simulation result. Overall comparison

Figure 3.18 shows all the results of our experiments briefly. Values that exceed the allowable range are shown in red. Note that the Automatic Gear Selection is within the allowable range for all experiments.

3.6. Discussion

As we expected, the original Pure Pursuit showed different results depending on the lookahead-distance. At a low speed of 4.5m/s, all the tested lookahead-distances

could be seen to be stable by showing a lateral Jerk within the acceptable range, but at a long 30m, the Distance error could be found to be greater than the acceptable range. Experiments on a regular curve without a transition curve showed that the distance error was greater than the allowable range even in the additional 20m lookahead-distance. This is where we can confirm that the original Pure Pursuit is more vulnerable to non-transition curves. The performance of path tracking and the trade-off of the lateral-jerk can be seen clearly when the vehicle is driving at high speeds on a non-transition curve. The path tracking performance is good in the order of 5m-10m-20m-30m, but in contrast, the lateral-jerk is large in the order of 30m-20m-10m-5m.

The results of our first proposal, Steering Gear Pure Pursuit, showed slightly better results than the original Pure Pursuit. Single Geared, Three Geared, and Six Geared all have satisfactory results within tolerance for distance errors. However, as expected, Single Geared showed a very large lateral Jerk at low speeds, and a larger value at high speeds, indicating that it would be better to use more than one Gear. However, when running at high speeds in non-transition curves, both Three Geared and Six Geared were found to have exceeded the acceptable range of the lateral-jerk. Our proposed Steering Gear Pure Pursuit certainly suggests that while following performance has improved, it is still insufficient to reduce the lateral-jerk.

However, the results of our second proposal, Automatic Gear Selection Pure Pursuit, were surprising. At both low and high speeds, the following performance and the lateral-jerk were within acceptable limits, showing very good results with and without transition curves. This confirms that Automatic Gear Selection Pure Pursuit is a

very robust new path-following controller that compensates for all the shortcomings of the original Pure Pursuit we identified.

3.7. Conclusion

In this paper, we compare the algorithms representatively used for path tracking and introduce Steering Gear Pure Pursuit, which can preserve the advantages of Pure Pursuit controllers, which are most used, and compensate for vulnerabilities. As previously discussed, the existing Pure Pursuit controller had path-following performance and inverse properties of lateral-jerk at the choice of lookahead-distance, which was a very difficult and sensitive part to identify. This result was satisfactory not only at low speeds but also at high speeds, and at both transition curves and non-transition curves, i.e., at any type of road. Therefore, we believe that this new Automatic Gear Selection Pure Pursuit will contribute significantly not only to autonomous vehicles currently being studied or tested in many ways but also to unmanned racing-related areas where the optimal adjustment of lateral-jerk at high speeds is essential.

3.8. References

[1] Bagloee, S.A., Tavana, M., Asadi, M. et al. Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. J. Mod. Transport. Vol. 24, 2016, pp 284–303.

[2] Pei-Sung Lin, Zhenyu Wang, and Rui Guo. Impact of Connected Vehicles and Autonomous Vehicles on Future Transportation. Bridging the East and West. American Society of Civil Engineers, 2016, pp 46-53

[3] Massar M, Reza I, Rahman SM, Abdullah SMH, Jamal A, and Al-Ismail FS. Impacts of Autonomous Vehicles on Greenhouse Gas Emissions-Positive or Negative? International Journal of Environmental Research and Public Health, 2021, 18(11):5567.

[4] A.T. Kearney. How Automakers Can Survive the Self-Driving Era. https://www.es.kearney.com/automotive/article?/a/how-automakers-can-survive-the-self-driving-era. Accessed July 22, 2021.

[5] McKinsey & Company. Automotive revolution - Perspective towards 2030. https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/disruptive-trends-that-will-transform-the-auto-industry/de-DE. Accessed July 22, 2021.

[6] Consumers Report. Cadillac's Super Cruise Outperforms Other Driving Assistance Systems. https://www.consumerreports.org/car-safety/cadillac-super-cruise-outperforms-other-active-driving-assistance-systems/. Oct. 28, 2020. Accessed July 22, 2021.

[7] Todd Litman. Autonomous Vehicle Implementation Predictions. Implications for Transport Planning. Victoria Transport Policy Institute, 2017.

[8] D. H. Shin, S. Singh, and J. J. Lee. Explicit Path Tracking by Autonomous Vehicles. Robotica, vol. 10, no. 6, 1992, pp. 539-554.

[9] Mišel Brezak, and Ivan Petrović. Path Smoothing Using Clothoids for Differential Drive Mobile Robots. IFAC Proceedings Volumes, Vol. 44, Issue 1, 2011, pp 1133-1138.

[10] Richard Wallace, Anthony Stentz, Charles Thorpe, Hans Maravec, William Whittaker, and Takeo Kanade. First results in robot road-following. Proceedings of the 9th international joint conference on Artificial intelligence, Vol. 2 (IJCAI'85). Morgan Kaufmann Publishers Inc., 1985, pp 1089-1095.

[11] J. M. Snider. Automatic steering methods for autonomous automobile path tracking, Ph.D. Dissertation, Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08.

[12] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L. Jendrossek, et al. Stanley: The Robot That Won the Darpa Challenge. J. Field Robot. vol.23, no 9, 2006, pp 661-692.

[13] M. Buehler, K. Iagnemma, and S. Singh, The DARPA Urban Challenge: Autonomous vehicles in city traffic. Springer, Vol. 56, 2009.

[14] M. Buehler, K. Iagnemma, and S. Singh, The DARPA Grand Challenge: The greate robot race. Springer, Vol. 36, 2007.

[15] B. Paden, M. Cap, S. Z. Yong, D. S. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Transactions on Intelligent Vehicles, 1(1), 2016

4. PATH OPTIMIZATIONS FOR AUTONOMOUS GROUND VEHICLES CONTROLS

4.1. Introduction

The autonomous vehicle system is largely composed of Perception, Path planning, Decision maker, and Controller. Based on various sensor data from the perception module, the path planning module generates all the possible paths that the vehicle can travel and passes them to the decision making module, which selects the best route of these candidate routes and passes the final decided route to the controller. And finally, the controller uses various path following algorithms to calculate steering angles and acceleration and deceleration to follow that path. Among them, the controller plays a very important role in calculating the speed and steering value of the vehicle directly, such as the role of human arms and legs at the last position.

Academia and industries that study many autonomous driving systems have been constantly working on controllers that enable vehicles to follow a given path accurately and efficiently, resulting in many sophisticated controllers. These efforts are based on the goal of accuracy of vehicle path tracking, but they are additionally designed to prevent understeer or oversteer of the vehicle, and allow the minimum amount of lateral/longitudinal force to be derived. [13] For this reason, more complex calculations are required for controllers to consider more portions and produce optimal results. Therefore, if it is possible to share the role of the controller in other modules, such as

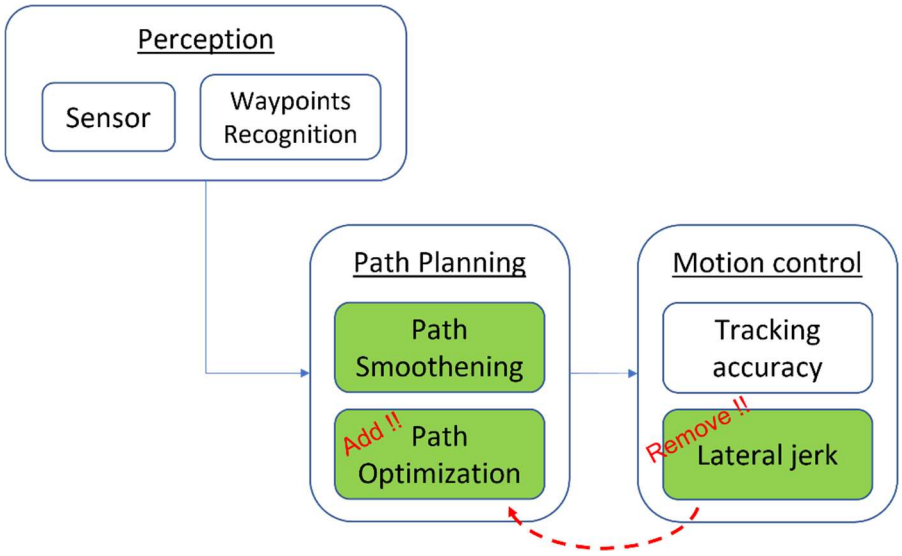


Figure 4.1 Autonomous vehicle's submodule that covers in this topic

path planning, it is possible to reduce the heavy burden on the controller and reduce the amount of computation.

However, in most studies, the path planning module is limited to generating only a drivable path and does not care how smoothly and comfortably the controller can follow the path. Furthermore, several path following algorithms published so far have not simultaneously satisfies path tracking performance and optimal ride comfort in any situation, suggesting that providing the optimal path for the controller during the path planning phase is the key to solving this problem. And we got the idea from the railways to solve this problem.

Trains follow only the perfect pre-designed railways, designed depending on the speed of the train and the curvature of the path. Unlike other modes of transportation, all railways are designed using transition curves to prevent railway derailment due to sudden changes in curvature, as trains cannot follow any other routes than the existing

railway. That is, following only the path of the transition curve, the train always can enter or escape certain curves smoothly without large lateral jerk. However, since cars are free to change paths by humans or computer systems, a sudden change of direction can lead to a larger lateral jerk, which in severe cases leads to an oversteer or understeer or cause multiple accidents.

A transition curve is a part of the curve that gradually increases or decreases at the beginning or end of the curve, thereby alleviating a sharp curve. The key to our proposed methods is to soften the path to the form of this transition curves so that the controller does not have to consider the lateral jerk when tracking the path while maintaining the path following accuracy and lesson the calculation. In addition, the ability to smoothen the path by eliminating perceived raw sensor data singularities in the path planning phase prior to making it a transition curve type is also an advantage.

In this paper, we explore the lane detection methods includes the curve fitting lane detection method, used in most existing studies, as a part of the path planning module, and address their weakness itself and what can be improved from the perspective of path planning to ease the burden on the controller. Finally, we propose methods using our new approach, discuss improvements from traditional methods through experiments through MATLAB simulation we implement and discuss the implications of these proposed methods. Through this, prove that our proposed methods are ways to broaden our horizons not only to adhere to lane detection's inherent function, but also to make it the easiest, simplest, and most reliable when tracking a path by the

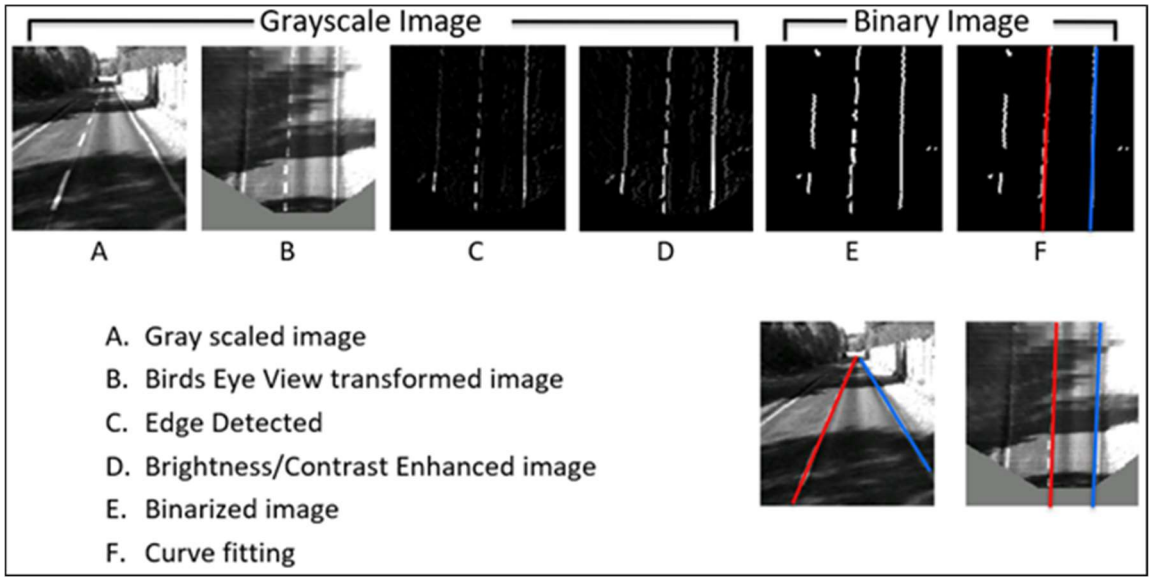


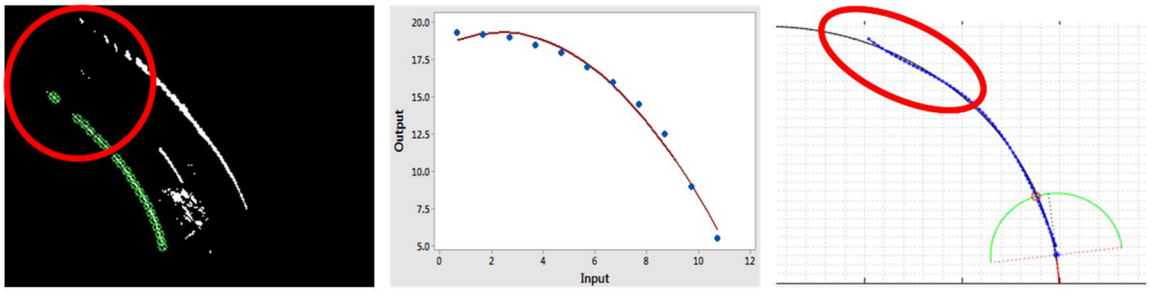
Figure 4.2 Traditional Lane Detection Method

controller, and show that we yield both low departure error and lateral jerk with the simple controller like basic Pure pursuit.

4.2. Background

4.2.1. Lane detection

Starting with self-driving car technology is lane detection that utilizes various sensors. A number of sensor fusion technologies such as camera and distance sensors are used to detect lanes, of which camera is the primary sensor. The tradition methods of lane detection techniques used in self-driving cars using cameras are grouping image gradients methods [1][2][3][4], or using Hough transform based on local edges extended image gradients [5][6][7][8], and recently, lane detection techniques based on deep neural networks have been studied [9][10][11]. Figure 4.2 shows a basic sequence of traditional methods in lane detection.



(a) (b) (c)

Figure 4.3 Curve fitting problems

In another sense, lane detection is often divided into two main categories. The first is to use the forward view taken over from the camera sensor as it is, and the second is to convert it to bird's eye view. There are pros and cons to each, but in the former case, there is no distortion of the shape because it uses the image as it is, but due to perspective, it is difficult to use the detected lane directly in a controller. Conversely, switching to bird's eye view can cause screen distortion over a long distance, but it is convenient for controllers in many ways because it allows them to align lanes as we view maps. Figure 4.2 also demonstrates the lane detection process using bird's eye view switching methods, and this study will also use bird's eye view switching methods.

In the phase of defining lanes after basic image processing, including denoising, many prior studies use a method of fitting curves with polynomials or spline curves.[7] In this case, the effect of softening raw data is great, but sometimes overfitting occurs[12], and the accuracy of partial detection of the lane may be somewhat reduced as fitting is concentrated only on the overall lane flow. (Figure 4.3) And most of the case, as Figure 4.3 (a) shows, the lane markings hardly detected on distanced area of the vision, so the longer distance area, the less accurate waypoints detection occurs. This

results in a large noise input in curve fitting, which inevitably results in inaccurate line detection results. In addition, it is recommended that the detected road can be expressed in a low-order fitting equation to represent it as smooth as possible, but this may lead to inaccuracy of detection and, on the contrary, overfitting as mentioned earlier.

More importantly, other studies so far show that there is no lane detection considering which routes the controller prefers, i.e., accuracy and comfort or safety in route tracking. All prior studies are devoted solely to lane detection's basic function, focusing only on how well it finds lanes.

4.2.2. Savitzky-Golay smoothing filter

There are several methods of smoothing signals. A typical smoothing method refers to a method of smoothing data by sequentially averaging a part of the entire data through windowing and displaying the average value as a representative value of the window, not the average of the entire data when the time series is listed in moving average. What can be quickly detected about the drawback of moving average here is that moving average uses the average value, which is well known to react very vulnerable to outliers. For this reason, some applications often use a median instead of an average value. Therefore, moving average has the advantage of being easy to implement, but it shows limitations that it is vulnerable to instant peaks. As one of the methods to compensate for this, there may be a method of smoothing by constructing a polynomial regression model for a short signal interval in the window that is applied to the time series.

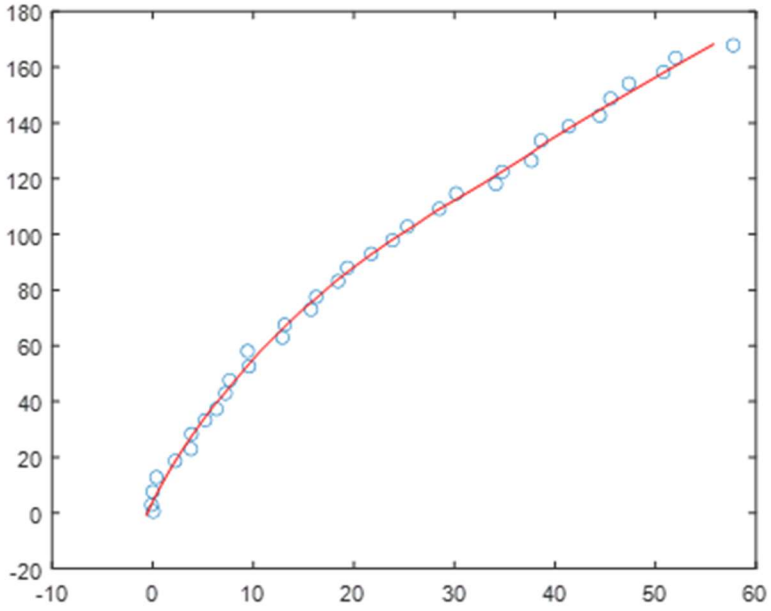


Figure 4.4 Savitzky-Golay filter test (frame length: 15)

The Savitzky-Golay filter [14], published in 1964, is a filter that finds a polynomial of k-order that best fits the surrounding points at each point in a least square way when there are given data, contain original information and noise, at regular intervals, and the result preserves the maximum, minimum, or peak width relatively well in a given data. The Savitzky-Golay filter suggests that in performing smoothing using regression model, it can accurately replace smoothing using a polynomial regression model by preparing a specific impulse response without calculating the regression model within the window of every time step. [15] In other words, if the properly calculated impulse response is used, the filter is designed to have the same effect as calculating a regression model for each window of every time step.

4.3. Methodology

The main idea of this topic is to make the raw data points (i.e., waypoints) given by the sensors a smooth curve with no major inflection, while allowing the controller to tracking this refined path very easily with having the low departure error and lateral jerk. It should be noted that all the methods we propose are start from receiving waypoints made through underlying image processing of raw data received from camera sensors. This corresponds to the E of Figure 4.2.

Unlike the traditional lane detection method which only performs curve fitting for smoothening the curve, we deal with the waypoints smoothening first, and then add additional curve optimization for the controller.

4.3.1. Curve smoothening

Prior to curve optimization for controllers, pre-processing for curve smoothing is required. Since we assume that the data taken over from the sensor had no special processing that was directly related, such as form modification of lane other than image processing includes basic denoising, the data taken over contains singularities due to other external factors such as solar reflection from the road. The process to remove this singularity and soften the detected lane.

4.3.1.1. Moving average of angle smoothening

At this stage, the key is to utilize the angle between each sequential point, instead of directly averaging the x and y points. We first calculate the difference between the x-

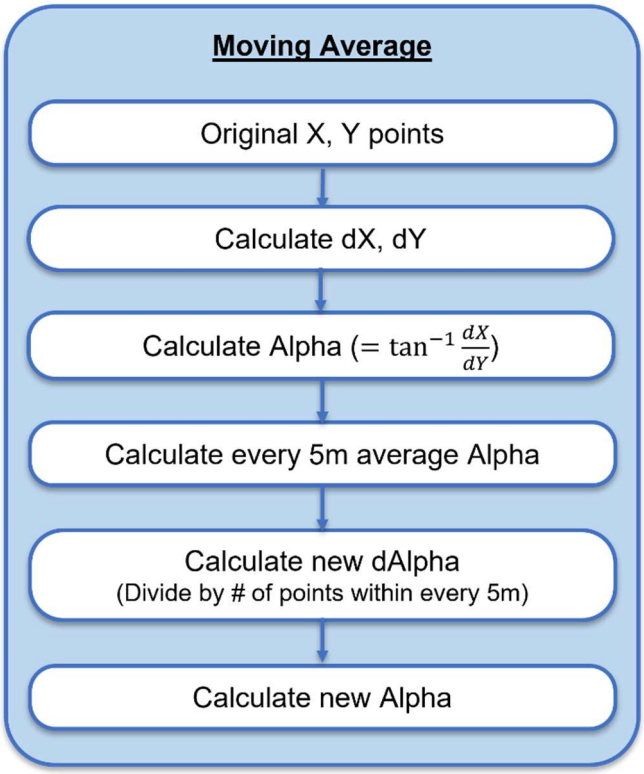


Figure 4.5 Moving average curve smoothening

and y-axis of each point, delta x(dx) and delta y(dy), to calculate the angle Alpha(α) between each sequential point. When Alpha is obtained, the mean of Alpha is obtained at intervals of 5 meters, and there is not a fixed number of points within each 5 meters. Therefore, the mean of Alpha is divided by the number of points contained within each 5-meter range and used as a modified Alpha for each point.

We performed moving average curve smoothening test for 30m length waypoints to check its performance, and the result shows 0.48m distance average error from the ground truth path and the angle of each waypoint does not smoothly change. (Figure 4.6) This still can be used since this unsmooth angle changes can be mitigated through the

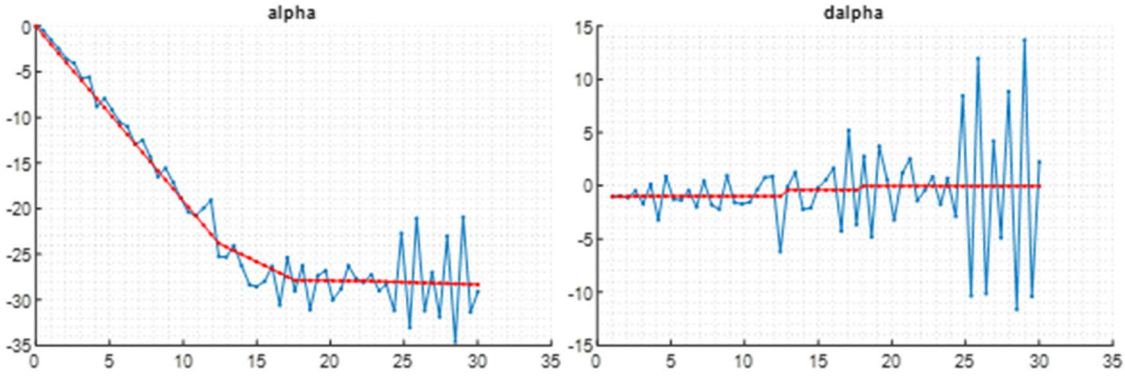


Figure 4.6 Moving average curve smoothening test

curve optimization, the next step, however, this issue can be solved with Savitzky-Golay filter which we will prove on 4.3.1.2.

4.3.1.2. Savitzky-Golay filter smoothening

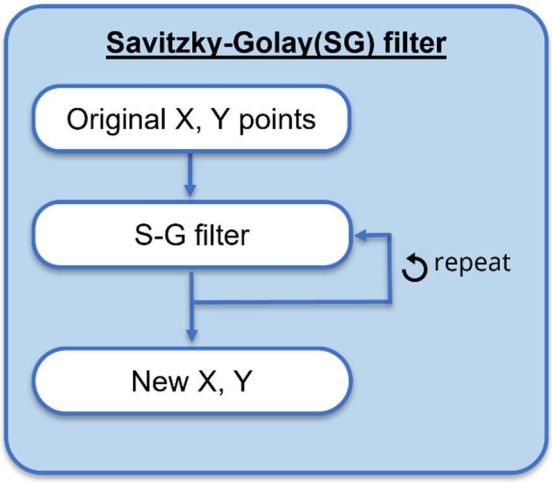


Figure 4.7 Savitzky-Golay filter curve smoothening

The Savitzky-Golay filter smoothing method is also receiving input of the size of the frame that processes it. It should be noted that we have set the frame size to 15. We used 30m as a typical driver's viewing distance, which is usually very rare to have more

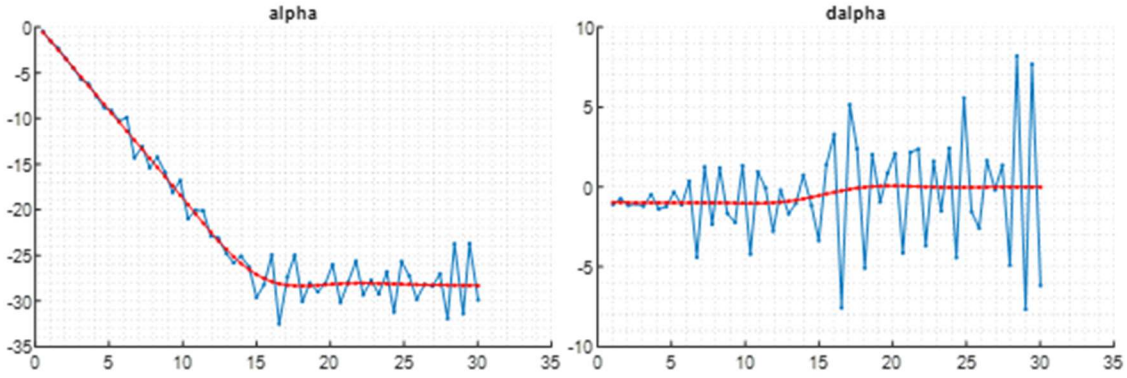


Figure 4.8 Savitzky-Golay filter curve smoothening test

than one large inflection point on the road within 30m, so it would be ideal to set the frame size to the number of all points within a distance of 30m. However, if there are two or more (multiple) inflection points on a road over 30m, smoothening's overperformance occurs, making the inflection points too dull. Therefore, in order to save the maximum information of the inflection point, we fixed the frame size to 15 (number of points within around 15m waypoints) and performed smoothing repeatedly.

As a result of testing Savitzky-Golay method under the same conditions as Moving average method, the Savitzky-Golay method shows better results than Moving average method with an average distance error of 0.12m. (Figure 4.8) In addition, the angle changes very smoothly. For this reason, the simulation was decided to proceed using Savitzky-Golay method as the curve smoothing in this paper.

4.3.2. Curve optimization for controller

Curve optimization for controller, a key procedure in this topic, is an additional process that eliminates significant changes in the curvature from the smoothened waypoints.

4.3.2.1. Curvature based optimization

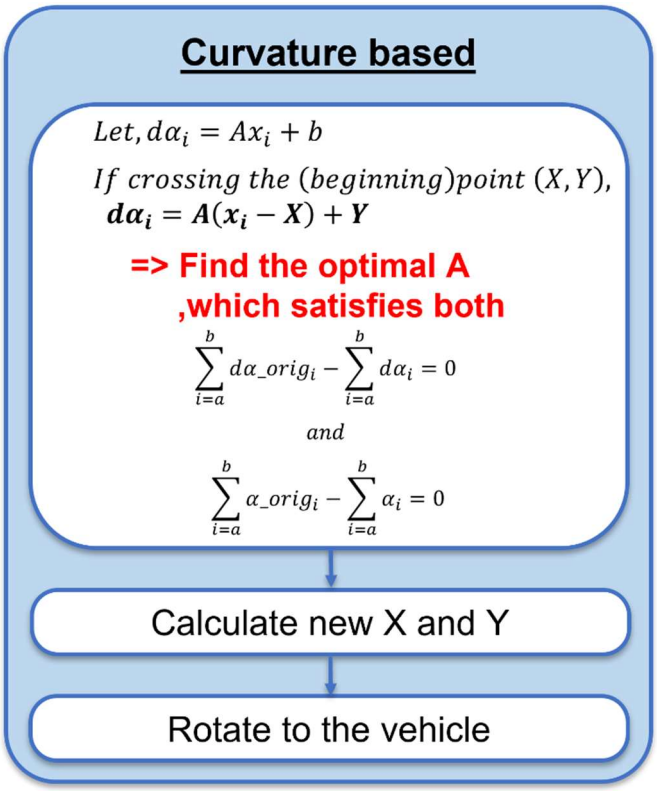


Figure 4.9 Curvature based optimization

Figure 4.9 shows overview of the Curvature based curve optimization. The key point of Curvature-based optimization is to make the sum of changes in angles and angles for the section of each point before optimization equal to the sum after optimization.

Let's first assume that the delta Alpha can be expressed in the following first order polynomials.

$$d\alpha_i = Ax_i + b$$

The above formula is valid if (0,0) is the starting point and the expression is expressed for the whole part, but the expression can be changed as follow to allow the whole part to be broken into parts and each to be optimized in order to maintain a more detailed representation.

$$d\alpha_i = A(x_i - X) + Y$$

At this time, our goal is to find the optimal slope A. First of all, the expressions sum of delta alpha and sum of alpha are as follows.

1) $Sum\ of\ delta\ Alpha$: $\int d\alpha$

$$= \sum_{i=a}^{b} d\alpha_i = \sum_{i=a}^{b} \{A(x_i - X) + Y\} = A\sum_{i=a}^{b}(x_i - X) + (b-a)Y$$

$$= A\textcircled{1} + \textcircled{2}$$

2) $Sum\ of\ Alpha$: $\int \alpha$

$$= \sum_{i=a}^{b} \alpha_i = \sum_{i=a}^{b} \left[\{A(x_i - X) + Y\}(x_i - x_{i-1}) + \sum_{j=a}^{i-1} \{A(x_j - X) + Y\}(x_j - x_{j-1}) + \alpha_{a-1} \right]$$

$$= \sum_{i=a}^{b} \left[A \left\{ (x_i - X)(x_i - x_{i-1}) + \sum_{j=a}^{i-1} (x_j - X)(x_j - x_{j-1}) \right\} \right.$$

$$\left. + Y \left\{ (x_i - x_{i-1}) + \sum_{j=a}^{i-1} (x_j - x_{j-1}) \right\} \right] + (b - a)\alpha_{a-1}$$

$$= \sum_{i=a}^{b} [A\{③ + ④\} + Y\{⑤ + ⑥\}] + ⑦$$

Since we aim to optimize the original curve while keeping the final direction unchanged, each Alpha and delta Alpha can induce the following expression by ensuring that the difference before and after optimization is zero.

$1 - 1)$ $Calculate\ optimized\ A\ only\ for\ dAlpha$

$$= \sum_{i=a}^{b} d\alpha\_orig_i - \sum_{i=a}^{b} d\alpha_i = 0$$

$$\Rightarrow dAl_{orig} - (A① + ②) = 0$$

$$\Rightarrow A = \frac{dAl_{orig} - ②}{①}$$

$2 - 1)$ $Calculate\ optimized\ A\ only\ for\ Alpha$

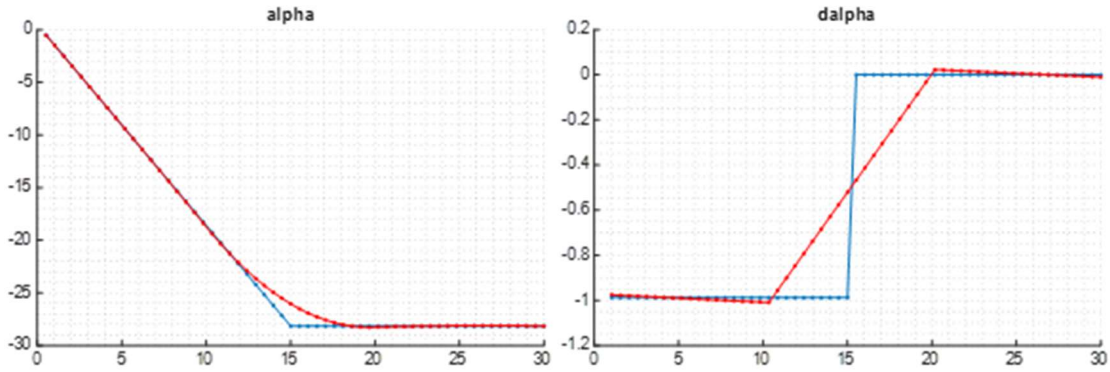


Figure 4.10 Curvature based optimization test

$$= \sum_{i=a}^{b} \alpha\_orig_i - \sum_{i=a}^{b} \alpha_i = 0$$

$$\Rightarrow Al_{orig} - [\{A(③ + ④) + Y(⑤ + ⑥)\} + ⑦] = 0$$

$$\Rightarrow A = \frac{Al_{orig} - Y(⑤ + ⑥) - ⑦}{③ + ④}$$

Use the two expressions above to find a common A, which is the optimized A,

3) $dAl_{orig} - (A① + ②) + Al_{orig} - [\{A(③ + ④) + Y(⑤ + ⑥)\} + ⑦] = 0$

$$\Rightarrow A = \frac{dAl_{orig} - ② + Al_{orig} - Y(⑤ + ⑥) - ⑦}{③ + ④ + ①}$$

We divide the entire interval into three parts to obtain each optimized A. The reason for dividing the section into three parts, as briefly stated above, is to maintain the original lane shape as much as possible. The more detailed the part is divided, the more

sophisticated the expression will be possible, but we decide not to divide it into too many sections because the amount of computation can be increased.

The test results were good as expected. (Figure 4.10) The average distance error was very small at 0.01m, and the angle change continued very smoothly. However, all calculations do not consider vehicle status such as location, heading, and previous steering angle. This is important since our purpose is to design the path which has all information of the vehicle so that vehicle just can follow without considering anything. Therefore, we add additional step for rotating entire optimized path to the vehicle's location.

4.3.2.2. Target-point based optimization

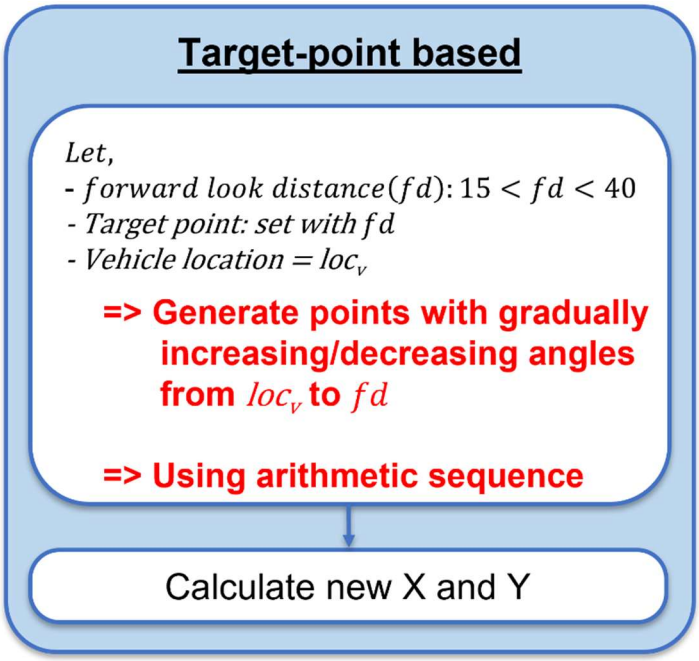


Figure 4.11 Target-point based optimization

Target-point based optimization is a method of producing points that gradually increase/decrease the angle corresponding to each point from current vehicle location to the fixed target point. To calculate this, the final point, the forward looking distance to obtain the final point, the location of the current vehicle, and the steering angle change in the previous situation are required.

The calculation can be much simpler than the Curvature based optimization. Since it is fundamental that the angle of each waypoint increases/decreases by the same difference, we start with the basic arithmetical progression formula.

$where,$

$a = \ angle\ change\ at\ the\ single\ waypoint$

$a_n = n^{th}\ term\ in\ the\ sequence$

$a_1 = the\ first\ term\ in\ the\ sequence\ (previous\ steering\ angle\ change)$

$N = number\ of\ waypoints$

$d = \ \ common\ difference\ between\ terms$

$S_n = arithmetic\ series$

$Loc_v = vehicle\ location\ (x, y)$

$P_f = final\ waypoint\ (x, y)$

$$a_n = \ a_1 + (n-1)d$$

Therefore, the total angle change within the given waypoints would be:

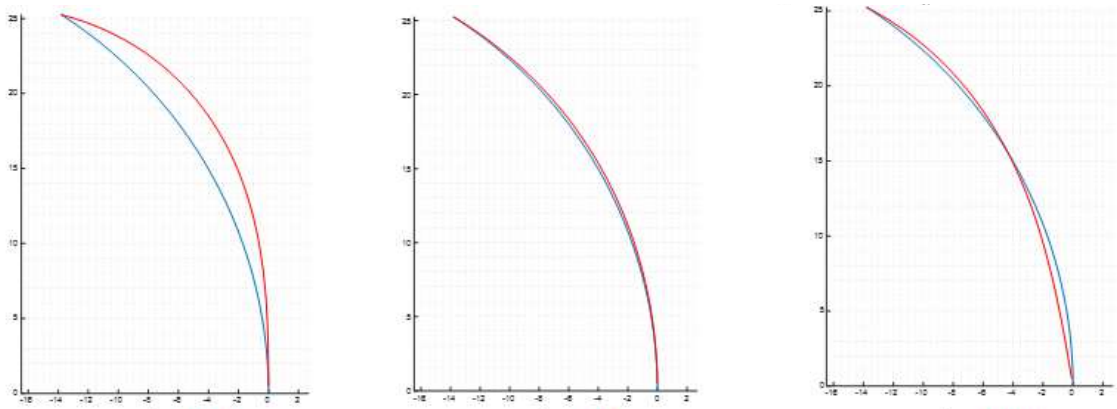


Figure 4.12 Curvature based optimization test

$$S_N = \frac{N(2a_1+(N-1)d)}{2} = \sqrt{(P_f(x) - Loc_v(x))^2 + (P_f(y) - Loc_v(y))^2}$$

And the common difference of angle changed *d* can be calculated with:

$$d = \frac{\frac{2S_N}{N} - 2a_1}{N-1}$$

After we get this value d, we can calculate the angle change at each waypoints, and then finally get the final X and Y of optimized waypoints.

As we expected, the optimized path has the same end point with the original path, and the angle differences changes smoothly increase/decrease. (Figure 4.12) Key advantage of this Target-point based optimization is that the optimized path already been designed with all information of the vehicle, so the controller just can follow this refined

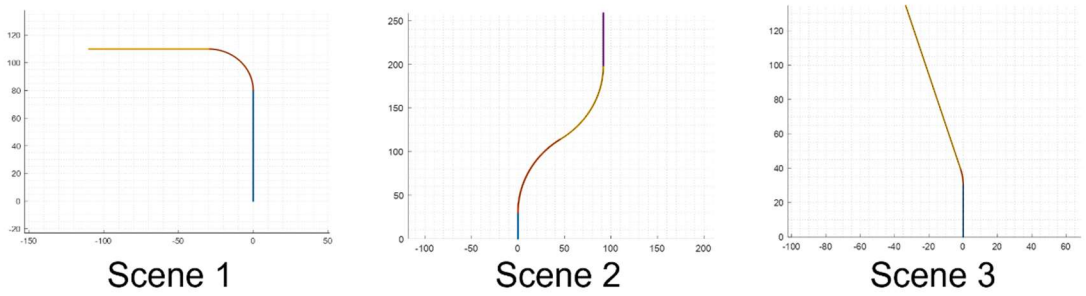


Figure 4.13 Simulation scenarios

path without considering anything about the path ahead and current vehicle's status. However, we should note that the performance is sensitive with the current vehicle's status as shown in Figure 4.12.

4.4. Experimental setting

We performed the simulation experiments with MATLAB. For the same conditions for all methods, we used fixed vehicle speed for 4.5m/s, steering angular speed for 0.4deg/ms, and operating frequency for 0.3sec, and used Pure pursuit for the path following controller. The lookahead distance for the Pure pursuit also fixed with the product of vehicle speed and operating frequency, however, the dynamic lookahead distance was used for simulating the Target-point based optimization algorithm, since this cannot be shorter than the forward looking distance. We set the lookahead distance as same as the forward looking distance if the forward looking distance is shorter than the default lookahead distance which is the product of vehicle speed and operating frequency. Three test scenarios are shown in Figure 4.13.

4.5. Result

4.5.1. Curve fitting

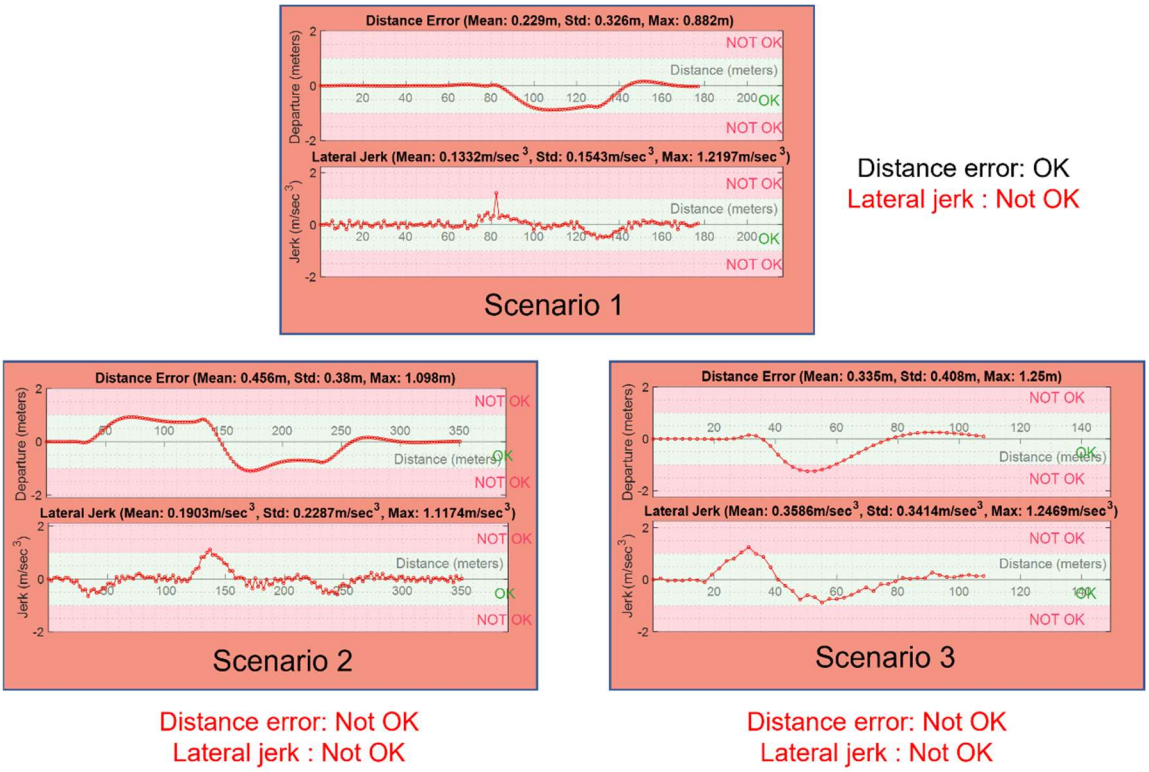


Figure 4.14 Simulation result - Curve fitting

As mentioned above, overfitting occurred in simulations using the traditional Curve Fitting method. This is the point of the sudden occurrence of a lateral jerk at 85m at our first scenario in Figure 4.14. No other unusual problem has been seen but given that the scenarios we experimented with consisting of simple curves rather than complex curves, more overfitting is expected if there is a lot of noise in signals accepted by the camera when dimming is poor.

4.5.2. Savitzky-Golay filter with Curvature based optimization

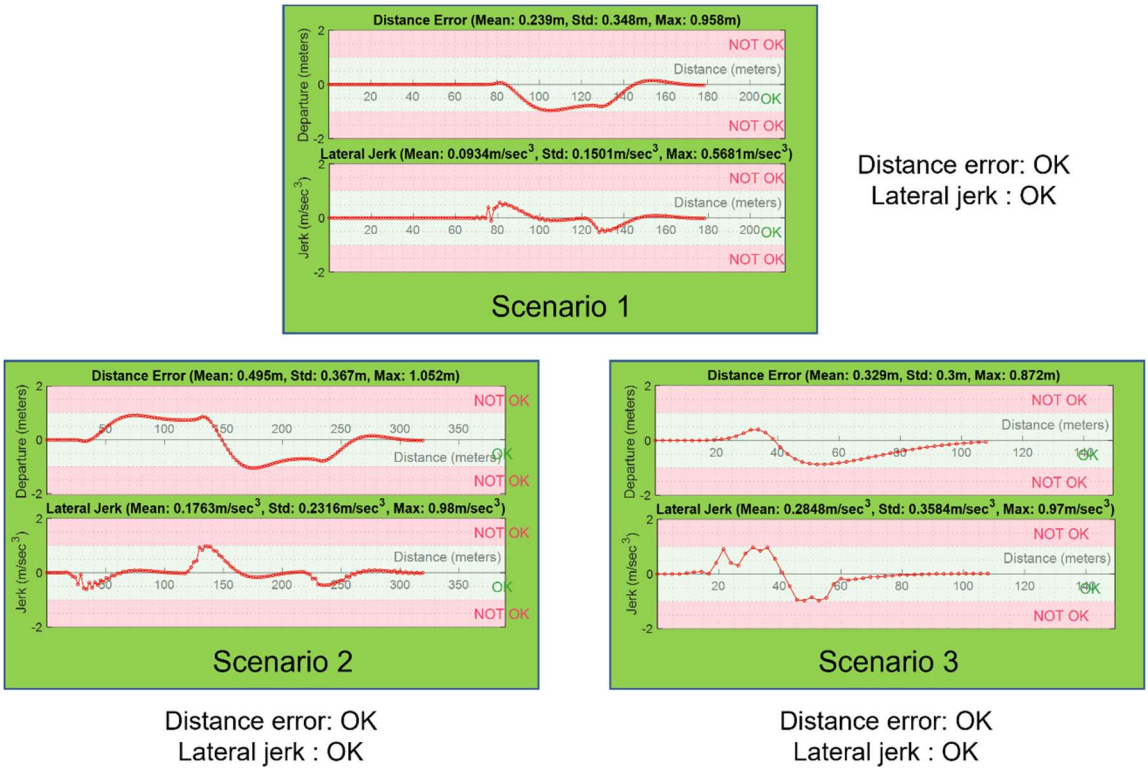


Figure 4.15 Simulation result - Savitzky-Golay filter with Curvature based optimization

Our first proposal, the simulation experiment of the Savitzky-Golay filter with Curvature based optimization method, did not cause any major problems. However, it can be seen that the lateral jerk is small but fluctuating so that it is not smooth. This is because the refined path does not contain information of the vehicle's heading. But as long as it's not over limitation and not jumping between + and -, it is still acceptable.

4.5.3. Savitzky-Golay filter with Target-point based optimization

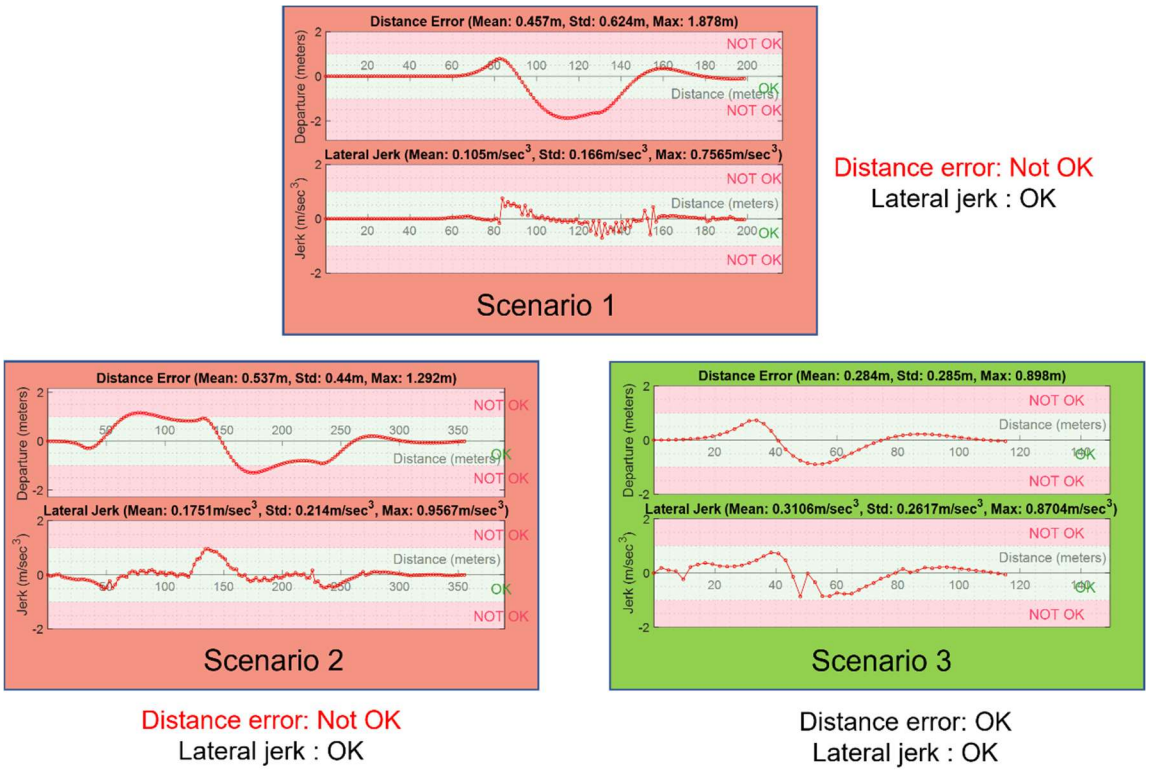


Figure 4.16 Simulation result - Savitzky-Golay filter with Target-point based optimization

Optimizing with Target-point based method shows lowest lateral jerk amongst all three simulation tests, however, has unacceptable distance error on both scenarios. Note that using short lookahead distance will have more distance error, opposite to the normal Pure pursuit controller's result. Since it optimizes the path which should have the same end point with the original path and gradually increasing/decreasing the changes of angle at the same time, more likely to moving closer to the original path when it uses shorter lookahead distance.

We managed to reduce the forward looking distance when the distance error is increased. The reason for this is that, in general, even when humans drive, if the vehicle deviates significantly from the lane, they look closer to the lane than they look far ahead

and re-enter the lane. In fact, a much larger distance error was identified without adding this function.

4.6. Discussion

		Curve Fitting	Curvature based Optimization	Target-point based Optimization
Scene 1	Distance error(m)	0.88	0.96	1.88
	Lateral jerk(m/sec^3)	1.22	0.57	0.76
Scene 2	Distance error(m)	1.10	1.05	1.29
	Lateral jerk(m/sec^3)	1.12	0.98	0.96
Scene 3	Distance error(m)	1.25	0.87	0.90
	Lateral jerk(m/sec^3)	1.25	0.97	0.87

Figure 4.17 Simulation result. Overall comparison

The experiments of the two novel methods proposed by this paper contain several important messages in that the optimal path to reduce the load of an autonomous vehicle controller could be provided in advance in the path planning stage. First of all, the Savitzky-Golay filter, which has not been used in other autonomous vehicle path smoothing studies so far, has been able to successfully remove noise and create a smooth path to the optimization stage. In addition, through the optimization of two methods through the curriculum analysis proposed by us, the later Jerk could be reduced. The redesigned path is, in other words, very similar to the controller's trajectory moving along the transition curve. Among the experiments we have performed, combination of Savitzky-Golay filter path smoothing and Curvature based optimization combination show that we can provide optimal driving Comfort with the most accurate path following

Curvature based optimization

		Speed = 4.5m/s	Speed = 2m/s
Scene 1	Distance error	0.97m	0.19m
	Lateral jerk	0.63m/sec³	0.15m/sec³
Scene2	Distance error	0.34m	0.06m
	Lateral jerk	0.39m/sec³	0.16m/sec³

Target-point based optimization

		Speed = 4.5m/s	Speed = 2m/s
Scene 1	Distance error	4.18m	2.85m
	Lateral jerk	0.79m/sec³	0.27m/sec³
Scene2	Distance error	2.18m	1.58m
	Lateral jerk	0.45m/sec³	0.28m/sec³

Figure 4.18 Performance based on vehicle speed

performance and the smallest lateral jerk among the experiments we have performed. And it should be noted that even though Target-point based path optimization method has larger lane departure, it is still a meaningful suggestion since the refined path contains the maximum information of the vehicle status and route ahead and it implies that there's no need for the complex controller design anymore. In addition, we can slow down the speed to mitigate the distance error for Target-point based optimization. Figure 4.18 show the result based on different speed and you can see that when lower the speed, we can get lower distance error and lower lateral jerk. And, this is not only for the Target-point based optimization, but also for the Curvature based optimization when it is needed. In the follow-up study, we can conduct how to combine the two proposed optimization methods which can provides the path for minimum distance errors and lateral jerk while carrying overall information of vehicle's condition and the road ahead, and add speed control.

4.7. References

[1] ZuWhan Kim. Robust lane detection and tracking in challenging scenarios. Transactions on Intelligent Transportation Systems, vol. 9, no. 1, pp. 16–26, 2008.

[2] Hendrik Deusch, Jurgen Wiest, Stephan Reuter, Magdalena Szczot, Marcus Konrad, and Klaus Dietmayer. A random finite set approach to multiple lane detection. International Conference on Intelligent Transportation Systems, pp. 270–275, 2012.

[3] Hunjae Yoo, Ukil Yang, and Kwanghoon Sohn. Gradientenhancing conversion for illumination-robust lane detection. Transactions on Intelligent Transportation Systems, vol. 14, no. 3, pp. 1083–1094, 2013.

[4] Tao Wu and Ananth Ranganathan. A practical system for road marking detection and recognition. Intelligent Vehicles Symposium, pp. 25–30, 2012.

[5] Byambaa Dorj and Deok Jin Lee. A precise lane detection algorithm based on top view image transformation and leastsquare approaches. Journal of Sensors, vol. 2016, 2016.

[6] Kamarul Ghazali, Rui Xiao, and Jie Ma. Road lane detection using h-maxima and improved hough transform. International Conference on Computational Intelligence, Modelling and Simulation, pp. 205–208, 2012.

[7] Yue Wang, Dinggang Shen, and Eam Khwang Teoh. Lane detection using spline model. Pattern Recognition Letters, vol. 21, no. 8, pp. 677–689, 2000.

[8] Bin Yu and Anil K Jain. Lane boundary detection using a multiresolution hough transform. ICIP, vol. 2, pp. 748–751, 1997.

[9] Bei He, Rui Ai, Yang Yan, and Xianpeng Lang. Accurate and robust lane detection based on dual-view convolutional neutral network. Intelligent Vehicles Symposium, pp. 1041– 1046, 2016

[10] Jun Li, Xue Mei, Danil Prokhorov, and Dacheng Tao. Deep neural network for structural prediction and lane detection in traffic scene. Transactions on neural networks and learning systems, vol. 28, no. 3, pp. 690–703, 2016.

[11] Jigang Tang, Songbin Li, and Peng Liu, A review of lane detection methods based on deep learning. Pattern Recognition, pp. 107623, 2020.

[12] Taskin Kavzoglu. Determining Optimum Structure for Artificial Neural Networks. Proceedings of the 25th Annual Technical Conference and Exhibition of the Remote Sensing Society, pp. 675-682, 1999.

[13] A Study of Lateral Vehicle Control Under a 'Virtual' Force Framework

[14] A. Savitzky and M. J. E. Golay, "Soothing and differentiation of data by simplified least squares procedures", Anal. Chem., vol. 36, pp. 1627-1639, 1964.

[15] R. W. Schafer, "What Is a Savitzky-Golay Filter? [Lecture Notes]," in IEEE Signal Processing Magazine, vol. 28, no. 4, pp. 111-117, July 2011.

5. CONCLUSIONS

It would be ideal to implement self-driving cars most similarly to human manipulation. However, it is not easy to fully implement human judgment and movements so far, so both academia and industry are spurring the development of self-driving cars, and in fact, research on self-driving is one of the hottest areas now. As part of such efforts, this study also conducted in various ways to satisfy the driving accuracy, safety, and comfort of the autonomous ground vehicle. In the first research topic, through risk detection using simple image signal processing that other studies have not yet thought of, we proposed an auxiliary early warning system for areas where sophisticated machine learning methods may miss and improved the area of 'Safety'. And in the second topic, a new type of Pure pursuit controller was proposed, and the performance of our system was verified by implementing our own simulation environment, and it showed the same or excellent performance as the existing traditional methods. At the last and third topic, a new Path planning method based on the idea from the train rail was proposed, which is noteworthy that the existing methods did not utilize the detailed characteristics of the designed road, but we introduced them into the autonomous driving system by utilizing the characteristics from the standpoint of designing the road and these last two topics covering the 'Driving accuracy' and 'Comfort' at the same time.

The first study of this study aims to enable passengers or autonomous driving systems to control speed through early warning or to avoid dangerous situations through steering. The second and third topics do not contain content about speed control, but

only about lateral control, which is intended to be mainly dealt with in this study, to make it clear that it is intended to deal with the controller's lateral control ability.

However, since the last two topics of this study did not conduct actual application experiments to self-driving cars and only derived results through computer simulation implemented with MATLAB, considerations for various external factors that will occur in the actual situation should be confirmed and applied through future studies. In addition, the simulation we conducted used a basic Bicycle model that is slightly different in motion from the actual four-wheeled vehicle, and also used a fixed steering angle rate value, so sufficient testing and verification must be performed to apply it to the actual vehicle.