creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

Ph.D. DISSERTATION

Robust Localization for Urban Autonomous Driving based on Road Environment Perception

도심자율주행을 위한 도로환경 인식기반의 강건한 측위 기법에 관한 연구

Soomok Lee

AUGUST 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Robust Localization for Urban Autonomous Driving based on Road Environment Perception

도심자율주행을 위한 도로환경 인식기반의 강건한 측위 기법에 관한 연구

Soomok Lee

AUGUST 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Robust Localization for Urban Autonomous Driving based on Road Environment Perception

도심자율주행을 위한 도로환경 인식기반의 강건한 측위 기법에 관한 연구

지도교수 서 승 우

이 논문을 공학박사 학위논문으로 제출함

2019년 8월

서울대학교 대학원

전기 컴퓨터 공학부

이 수 목

이수목의 공학박사 학위 논문을 인준함

2019년 8월

위 원 장:	최 진 영	
부위원장:	서 승 우	
위 원:	김 현 진	
위 원:	김 성 우	
위 원:	최 민 음	

Abstract

Automated driving vehicles or advanced driver assistance systems (ADAS) have been an important research topic in the field of transportation. This can promise to reduce road accidents and eliminate traffic congestion. Among them, highly accurate and robust real-time localization is a crucial part for various autonomous driving applications. Knowing the position of the ego-vehicle is the most fundamental and inevitable step in that it is not possible to begin automated driving unless the vehicle is localized. Numerous methods for localization have been proposed combining various kinds of sensors including environmental sensors, IMU and GPS. Among them, the usage of environmental sensors are core technology in that GPS cannot guarantee the ideal pose.

Majority of GPS systems indicate deteriorated data such as two or three lanes away from the ground truth position. Those inaccuracies are compensated by various kinds of environment sensors. The major concern for previous research is maximizing the accuracy and precision of the systems while minimizing the cost. However, most of these studies do not take into account the stability of the localization technology.

Motivated by this stability aspect, not only do we aim to present precise and cost-effective localization system and relevant perception algorithms, but also take a step forward through the suggestion of a way of ensuring stability of vehicular localization in this dissertation. To achieve this goal, the proposed three algorithms which include visual perception of road environments, visual localization using perception result and road network map, and its expansion schema ensuring stability as a redundancy for fully self-driving vehicles.

First of all, localization based on a single map matching source occasionally fails due to insufficient features from innate fragile moments. Other matching source can be augmented to the original localization system as a redundancy in pursuit of stability.

To achieve this goal, we propose a multi-modal localization framework using heterogeneous map-matching sources. The localization framework utilizes two independent map-matching sources and integrates them with a stochastic situational analysis model. Adapting a stochastic model calculates the reliability of the two map matching sources and solves the fail-safe problem through reliability-based system stability. A number of experiments with autonomous vehicles within the actual driving environment have shown that combining multiple map-matching sources ensures more robust results than the use of a single environmental sensor.

In addition, we propose an ego-lane index aware lane marking based localization system in chapter 2. In terms of proposing multi-lane detection and ego-lane index identification in a single integrated deep learning network, we guarantee localization accuracy under ensuring real-time processing conditions. The effect of the perceived lane index information on the stability of the lane-based localization system is proven by demonstrating various types of harsh condition.

Lastly, the basis of these localization algorithms should be that the stability of the fundamental road marker detection performance is maintained. In chapter 4, we propose the line-segment-level clustering method for properly exploiting both lane and parking-line. The line segments are properly extracted with the proposed Directional-DBSCAN line-instance-level clustering algorithm. Parking lines and multi-lane detection is applied to demonstrate robustness of the proposed clustering method. In comparison to other line detectors, we show that the Directional-DBSCAN algorithm robustly extracts road lines and enhances the localization performance even when they face sophisticated lines such as short, curved, merged or split road-line.

Overall, the research presented in the dissertation discusses how visual sensor can be effectively used to understand road-network-level environments and how those extracted information is applied to robust localization. Moreover, the presented pipeline is designed to integrate heterogeneous map matching sources using the enhanced road-

network map matching methods. This pipeline ensures localization stability and this has been proved within various urban road scenarios.

keywords: Automated driving, vehicle localization, sensor fusion, machine learning, vision processing, road environment perception

student number: 2013-20850

Contents

Abstract **i**

Contents **iv**

List of Tables **viii**

List of Figures **x**

1 Introduction **1**

1.1 Background and Motivations 1

1.2 Contributions and Outline of the Dissertation 3

1.2.1 Fail-Safe Multi-modal Localization Framework using Multiple Map-matching Sources 4

1.2.2 Ego-lane Index-aware Vehicular Localization using DeepRoad Network for Urban Environments 5

1.2.3 Directional-DBSCAN: Road Marking Line Instance-level Clustering Method for Parking Slot and Lane Detection 5

2 Fail-Safe Multi-modal Localization Framework using Heterogeneous Map-matching Sources **7**

2.1 Introduction 7

2.2 Framework 11

2.2.1 Sensor Configuration 11

2.2.2 Localization Framework 12
2.3 Preprocessing for Road Network Map Matching 12
2.3.1 Lane Detection 14
2.3.2 Selective Reading of Road Network Map 14
2.4 Heterogeneous Map Matchings 14
2.4.1 Road Network Map Matching 15
2.4.2 Point-cloud Map Matching 16
2.5 Probabilistic Matching Importance Analysis 17
2.5.1 Status Quantification: Overview 18
2.5.2 Reliability Quantification of maps 19
2.5.3 Reliability Quantification of Sensors 23
2.5.4 Matching Reliability-Weight 27
2.6 Fail-safe Status Analysis 30
2.6.1 Matching Consistency Analysis 30
2.6.2 System Instability Response 31
2.7 Localization Model 32
2.7.1 Motion Update 32
2.7.2 Integrated Measurement Update 32
2.7.3 State Estimation and Resampling 33
2.8 Experiments and Evaluation 33
2.8.1 Experimental Set-up 33
2.8.2 Experiments & Evaluations 37
2.9 Conclusion 44
2.10 Appendix 45

3 Ego-lane Index-aware Vehicular Localization using DeepRoad Network for Urban Environments 46

3.1 Introduction 46
3.2 Related Works 48

3.2.1 Lane Detector 49
3.2.2 Lane-index Identification 50
3.2.3 Lane marking based localization system 51
3.3 Framework 52
3.4 DeepRoad-Net: Lane Marker Detection 52
3.4.1 Network structure 53
3.4.2 Data Augmentation 53
3.4.3 Implementation Detail 54
3.4.4 Adaptive Birds Eye View Transform 55
3.4.5 Lane Marker Feature Clustering and Fitting 56
3.5 Ego-lane index Identification 57
3.5.1 Primal Input Data for the Architecture 57
3.5.2 Ego-lane Index Identification: Network Architecture 58
3.5.3 Data Augmentation 61
3.5.4 Implementation Detail 61
3.6 Localization using Extracted Lane and Lane-index Information 62
3.6.1 Local Positioning 62
3.6.2 Global Positioning: Road Network Map Matching 64
3.7 Experiment and Evaluation 67
3.7.1 Experimental Set-up 67
3.7.2 Evaluation on Lane Detection Performance 68
3.7.3 Evaluations on Ego Lane-index Identification 70
3.7.4 Evaluation on localization performance 74
3.8 Conclusions 78

4 Directional-DBSCAN: Road-line Instance Clustering Method for Parking Slot and Lane Detection 79

4.1 Introduction 79
4.2 Road Marking Association using Directional-DBSCAN 83

4.2.1 Pre-process for setting Around View Monitoring System . . . 84
4.2.2 Road Marking Feature Extraction . . . 84
4.2.3 Road Marking Clustering using Directional-DBSCAN . . . 85
4.2.4 Line fitting of Line Marking Clusters . . . 90
4.3 Slot Pattern Recognition and Detection . . . 90
4.3.1 Pre-Processing for slot-pattern Recognition: Adjacent parking slot grouping . . . 91
4.3.2 Slot Pattern Classifier . . . 92
4.3.3 Parking-slot Detection . . . 94
4.4 Experiments and Evaluation . . . 95
4.4.1 Experimental Set-up . . . 95
4.4.2 Evaluation . . . 95
4.5 Conclusion . . . 103

5 Conclusion **104**

Bibliography **106**

Abstract (In Korean) **117**

List of Tables

2.1 Technology migration in sensor fusion based localization. 9
2.2 Computational cost at each step 37
2.3 Localization Performances in Yeouido : Mean (Standard dev.) 39
2.4 Localization Performances in Palo Alto : Mean (Standard dev.) 40
2.5 LIDAR Reliability Weight (γ): Mean (Standard dev.) 44
2.6 LIDAR Reliability Weight in different road scenarios (γ) 44

3.1 Technology migration in lane-level localization. 50
3.2 DeepRoad-Net architecture 54
3.3 Index identification network architecture 60
3.4 Performance Comparison in Various Network Structures. 69
3.5 Performance Comparison to Other Detectors: KITTI Dataset 70
3.6 Ego-lane index classification accuracy of various identifiers 72
3.7 Lane-index identification in various weather conditions 73
3.8 Performances of each lane driving conditions (Accuracy). 74
3.9 Localization Performances: Mean (Standard dev.) 75

4.1 Technology migration in parking-slot detection. 81
4.2 Performance evaluation in line-segment clustering. 96
4.3 Performance comparison between different modules for its detection purpose. 97

4.4 Slot-detection performances on diverse environments. 99
4.5 Localization performances: Mean (Standard dev.) 102

List of Figures

1.1 Core technologies for automated driving. 2

1.2 Migration of the proposed methods under the original map matching framework. 4

2.1 Feature cases on point cloud's height map matching 8

2.2 Localization Framework 10

2.3 NDT of point cloud 16

2.4 Suffciency score function graphs 19

2.5 Occupancy function graphs 21

2.6 Examples of Map belief 22

2.7 Examples of sensor belief 25

2.8 Measurement reliability function $m(x_j)$ 27

2.9 Dissimilarity detection of multiple matching sources. 30

2.10 Two types of maps for trajectory B. 34

2.11 Localization results at Yeouido (shown in road network map). 36

2.12 Localization results at Palo Alto (shown in point cloud map). 38

2.13 Autonomous vehicle test platform 38

2.14 A scenario of localization result 40

2.15 Localization Performance at a certain interval. 41

2.16 Localization performance comparison in artificial errors. 45

3.1 Ego-lane estimation network 51
3.2 DeepRoad-Net 55
3.3 Input and outputs of each step in the Ego-lane Identification Network 58
3.4 Input of ego-lane sequence images: case of driving in the second lane 58
3.5 Map-matching lane 64
3.6 Measurement scores($w^N_{Lane,k}$) of given pose candidates ($\mathbf{x}^N_k$) whether ego-lane index information is given or not. Pose (0,0) presents the actual pose. Black line presents local lane maps. 65
3.7 Lane detection results 70
3.8 Ego lane results 71
3.9 Lane and Ego-lane index results from various dataset scenarios: (a) Shiny, (b) Cloudy, (c) Night, (d) Rainy conditions 73
3.10 Yeouido Datatset. 75
3.11 Ego-lane index-aware localization performance within intersection region 76
3.12 Errors of the localization results in Figure 3.11 LaTeX Error: Can be used only in preambleSee the LaTeX manual or LaTeX Companion for explanation.Your command was ignored.Type I ¡command¿ ¡return¿ to replace it with another command,or ¡return¿ to continue without it.3.11 76

4.1 Difficulties in parking-line detection on distorted, short, and curvy lines: With same marking features (black), (a) Probabilistic HT detector (green)[65] missed many short lines, and long curved lines are snapped compared with (b) proposed D-DBSCAN results. (c) Corner detector[69] is also weak at blurry and distorted angles. 82
4.2 From left to right: DBSCAN clustering, D-DBSCAN clustering, line fitting results of D-DBSCAN clusters. 90
4.3 Two-step classifier 92
4.4 Directional-DBSCAN clustering results in the dataset. 96

4.5 Lane marking clustering of feature points using Directional-DBSCAN. 98
4.6 Example of proper lane-instance level clustering in top view perspective. 98
4.7 Example of proper lane-instance level clustering in front view perspective. 99
4.8 Test results in diverse types of slots(simple rectangular/diamond(1st), diamond with double lines/open(2nd), open slanted/parallel(3rd), rectangular distortion on unflat terrains(4th), slanted rectangular(5th). . . 100
4.9 Test results in diverse environments scenarios (underground, fallen leaves, snowy, rainy and sunset). 100
4.10 Test results in various difficult environments (co-existence of pillar, vehicle occlusion, brick lines, curb and irrelevant types of road markers.) 100
4.11 Detection of parking moment after selecting parking space. 101
4.12 Localization result in the target zone. 102

Chapter 1

Introduction

1.1 Background and Motivations

Automated driving is becoming a familiar concept. Technologies related to the automated driving, such as lane keeping or smart cruise control, have been spread to the public, particularly in highway driving conditions. As a result, safety and efficiency of the practical driving are expected improve gradually. With the development of self-driving system, the need for automated door-to-door service arises. Because of this, the need for self-driving is also expanding from highway environment to complex urban and other difficult environments. However, for the ideal automated driving that can achieve a door to door driving, various element technologies, such as map building, localization, perception, and planning & decision making, should be developed with higher accuracy and reliability as shown in Fig. 1.1.

Among them, highly accurate, robust and real-time localization is of great importance for various autonomous driving applications. Knowing the position of the ego-vehicle is the most fundamental and inevitable step in that it is not possible to begin automated driving unless the vehicle is localized in a centimeter-level accuracy. Conventional approaches for lateral localization utilize some combined information from a high-precision Global Positioning System (GPS), an inertial measurement unit

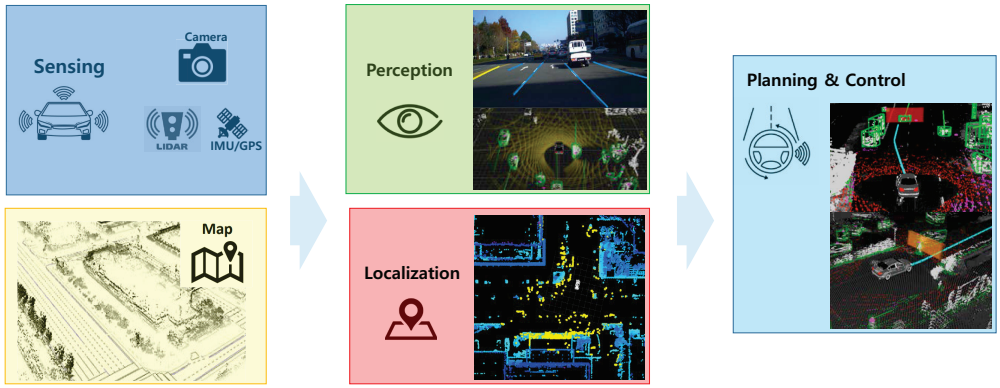


Figure 1.1. Core technologies for automated driving.

(IMU) and a digital map. Typical GPS receivers mounted in consumer vehicles provide rough information on the vehicle location with 5 to 10 meters deviations from the real position, which is not sufficient to attain the goal of this study. Especially in case of the complex urban driving for the aforementioned needs, poor satellite geometry and disruption of radio signal reception due to skyscrapers further lower the accuracy of vehicle pose.

To overcome those inaccuracies, simultaneous localization and mapping (SLAM) system derives relative poses and performs pose refinements such as loop closure optimization after accumulating a certain amount of poses. The system not only requires a substantial computation cost, but also contains partially accumulated pose errors in local positioning until performing pose refinements. This means that those limitation are not suitable for the real-time localization especially in automated driving platforms even though it works well in building a map as a post-processing. To fulfill both real-time and high accuracy within a global pose, map matching based localization between environment sensors and corresponding maps has been more reasonable solution. In previous researches, major concern of environment sensor based localization is maximizing the accuracy and precision of the systems while minimizing the computational cost. In order to fulfill requirements on lower cost localization system, introduction of enhanced detection with lighter features for efficient map matching methods has

been proposed. However, most of these studies do not take into account the stability of the localization technology. Motivated by this stability aspects, not only do we aim to present precise and cost-effective localization system enhancing relevant perception algorithms, but we also take a step forward through the suggestion of a way of ensuring system stability of vehicular localization in this dissertation.

In this dissertation, the proposed three algorithms which include road environments perception, road network map matching based localization using perceived outcomes and road network maps, and multiple map matching sources based localization framework to ensure stability as redundancy for fully self-driving vehicles. First of all, we propose a multi-modal fusion based localization system using multiple map matching sources. This fusion framework maximizes the use of each map matching measurement, ensuring the stability of the system and maximizing the effect in terms of the redundancy. Secondly, we propose machine learning and vision based road environment recognition algorithms such as road marking, lanes and ego-lane index which allows local pose estimation systems. We show that lane index aware-localization system using those detection results can improve performance as well as improving system stability. In addition, we propose the road marking instance clustering methods for road environment such as lanes and parking lines. The proposed algorithm is meaningful in that enhanced road marker instance clustering allows the road network map based matching system more accurate.

1.2 Contributions and Outline of the Dissertation

Framework, Fig. 1.2, describes how each chapter is positioned to the backbone particle filter based map matching system. Road network map based semantic feature matching is advance by proposing ego-lane index identification from chapter 3 and enhanced lane detection with proper instance-wise clustering from chapter 4. After enhancing stability of semantic map matching, we proposed how the semantic map matching can

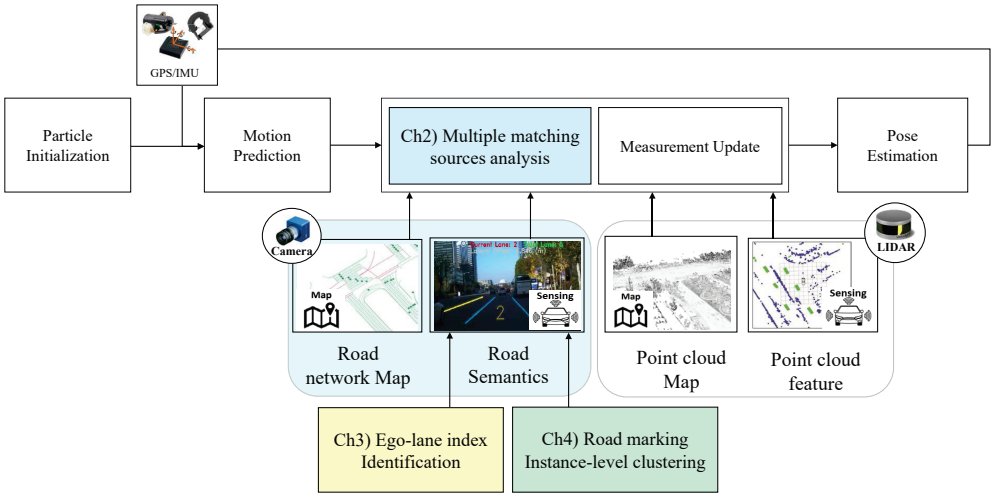


Figure 1.2. Migration of the proposed methods under the original map matching framework.

complement with other matching source based map matching in chapter 2.

1.2.1 Fail-Safe Multi-modal Localization Framework using Multiple Map-matching Sources

High accurate and robust real-time localization is a crucial technique for various autonomous driving applications. Numerous methods for localization have been proposed combining various kinds of sensors including an environmental sensor, IMU and GPS. However, the usage of a single environmental sensor is rather fragile. Although the use of multi-environment sensors is a better alternative, fusion methods from previous studies have not adequately compensated for shortcomings in dissimilar sensors or have not considered errors in the pre-built map.

In this paper, we propose a multi-modal fusion-based localization framework using multiple map matching sources. The framework contains two independent map matches and integrates them with stochastic situational analysis model. By applying a stochastic model, the reliability between the two map matches is collected and the sys-

tem stability is verified. A number of experiments with autonomous vehicles within the actual driving environment have shown that combining multiple map matching sources ensure robust results than the use of a single environmental sensor.

1.2.2 Ego-lane Index-aware Vehicular Localization using DeepRoad Network for Urban Environments

Accurate localization of an ego-vehicle is one of the core technologies for autonomous driving. Among localization methods, camera sensor based methods have been attractive because of its smaller market entry barrier compared to LIDAR sensor. Among camera based localization, conventional approaches have utilized GPS data, prebuilt map information, and lane detection results to estimate location of an ego-vehicle. However, these approaches demonstrate performance limitations due to deteriorated data from GPS indicating two or three lanes further away, and identical shape of detected lane cues which makes unconscious of wrong lane status. This limitation prevents handling of various tasks in diverse driving environments.

In this paper, we propose an ego-lane index-aware lane based localization framework that utilizes road markers and ego-lane index to overcome these limitation. We also propose integrated DeepRoad-net which solves road marker detection and ego-lane index identification at the same time. We show that more accurate and reliable lane map matching localization results can be achieved by combining ego-lane index identification, which increases confidence and reliability of the results. We verify that our approach can be applicable to various environments including wide/narrow complex urban driving environments.

1.2.3 Directional-DBSCAN: Road Marking Line Instance-level Clustering Method for Parking Slot and Lane Detection

Parking slot detection or lane detection algorithms using visual sensors have been required for positioning ego-vehicle both locally and globally. In most previous studies,

popular feature association for detecting parking lines and multi-lanes, such as the Hough line detector or DBSCAN algorithm, have been employed. However, these algorithms are originally designed to find distinct features and are inadequate for the short, curvy, faint and distorted parking lines of long-range surround-view images. Similarly, lane feature clustering algorithm such as DBSCAN is ineffective in merging or split lanes. In this paper, we propose a robust line-segment-level clustering method for road marking detectors. The proposed algorithm consists of line-segment detection with the proposed Directional-DBSCAN line-level feature-clustering algorithm. Slot detection method is further proposed. In comparison to other feature detectors, we show that the Directional-DBSCAN algorithm robustly extracts lines even when they are short, faint, merging or splitting. Moreover, we verify that the improved lane detection algorithm can be applicable to enhancing lane-level localization by experiments on abundant dataset.

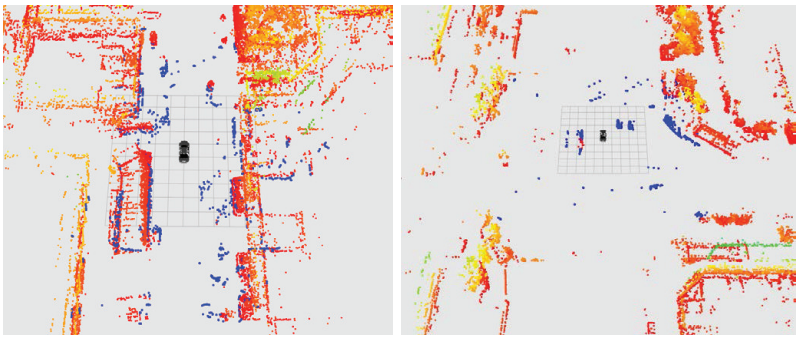
Chapter 2

Fail-Safe Multi-modal Localization Framework using Heterogeneous Map-matching Sources

2.1 Introduction

Localization is a technique for acquiring the position information of a vehicle while driving on a map. The technique is crucial for various autonomous driving modules such as global/local path planning, and map information utilization. For this, high-accuracy and fast-driven vehicle pose information should always be provided. Among various kinds of localization methods, map matching approaches for localization utilize some combined information from a high-precision Global Positioning System (GPS), an inertial measurement unit (IMU), environment scanning sensors and a digital map. With prior knowledge of the driving environments, these approaches show considerable localization performance.

Among map-matching based localization methods, a dense point-map matching method using environment sensors and dense map has been a good option for sustaining accuracy of the localization. Among the environment sensors, stereo camera [47][48] or LIDAR map matching methods [49][50] have been shown accurate performance due to its accurate distance results. However, 3D point matching is vulnerable in



Narrow road Wide road

Figure 2.1. Feature cases on point cloud's height map matching

high-traffic environments or wide road, as the extracted 3D features are mostly ineffective moving objects. In addition to the sensor limit, the 3D map matching performance can be easily degraded by map sensitivity coming from environment changes. An example for point clouds fragility in wide road condition is shown in Fig. 2.1. In the figure, the center grid in the figure is the pose of the vehicle. Reddish color presents map whereas blue color indicates current sensing point clouds. Because of lesser point clouds reflected from static objects, map matching scores are unreliable in a broad road scenarios.

On the other hand, road network map matching methods have been steadily introduced due to their simplicity and practicality, such as road markings and lanes. It is a good alternative for 3D dense map matching sources because road network maps are less likely to change and do not require frequent update. These road network features are normally extracted by camera sensors since the camera's information is much richer than the LIDAR's light reflection or other environmental sensors. However, road network map matching algorithms are usually confined to well-built roads. This type of matching also usually only ensures suboptimal levels of longitudinal accuracy, due to the intermittent presence of longitudinal markings and a slightly inaccurate camera distance measurement.

Fusion methods can complement each sensor's mismatch or characteristic weak-

Table 2.1. Technology migration in sensor fusion based localization.

Sen. / Map	Single Env. Sensor	Multi-Environment Sensors	
		Integrated	Parallel
Single Map	Not a fusion category [47]-[50]	CAM&SONAR[57] LIDAR&CAM[54][56]	CAM&LIDAR[55] LIDAR&AVM[51]
Multi Maps	2 LIDAR Maps LIDAR Sensor [53][58]	Not Found	Proposed

ness, and increase the reliability of the overall system. As shown in table 2.1, fusion have been developed in different ways depending on the combination of multiple sensors and maps. First, multiple environmental sensors and a single map are used for fusion. They are categorized by integrated and parallel processing method. In terms of decentralized method, Lee *et al* [51] suggested that LIDAR and camera are computed in parallel and adding them up as combined measurements. Zhang *et al* [55] designs the sequential fusion system that one sensor provides candidate region group and other sensor refines the pose after that. Finding candidate region group reduces the overall processing time to refine the accurate pose, but it can cause serious malfunction when the first approximation is failed. This eventually increases the dependency of a particular single sensor.

For integrated methods [54][56][57], multiple sensors are integrated as a single measurement set and matched with a single map. This improves the sensing coverage and enriches matching features. However, all these approach are greatly influenced by the accuracy of a single map. If the single map character is monotonous or contains slight error, the map matching result is vulnerable.

Meanwhile, Wan *et al* [53] suggest that a single LIDAR sensor is matched with multiple maps. The sensor utilizes the point intensity to match the 2D intensity map and the height (z) values to the 3D height map. Intensity matching is utilized to obtain

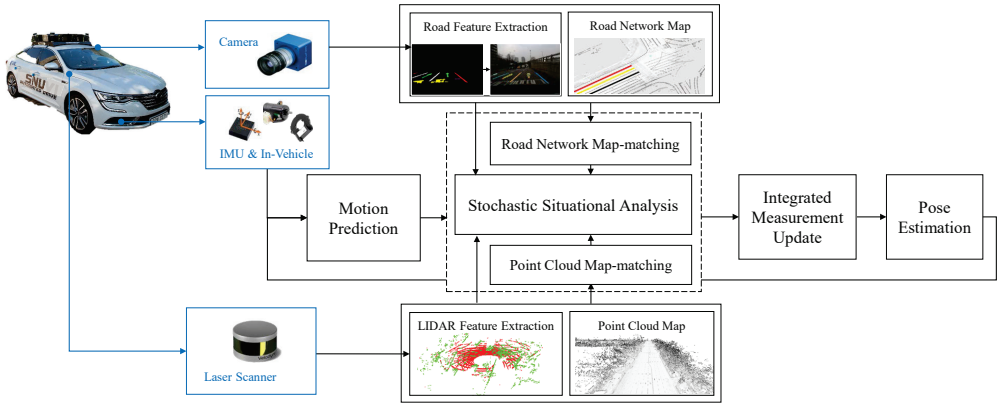


Figure 2.2. Localization Framework

approximate orientation and height map matching is for estimating the position values. Similar to the Wan's method, Kim *et al* [58] also adopted in the same map-sensor configuration. The redundancy of matching is similarly realized using intensity matching and height matching in single LIDAR sensor and multiple LIDAR maps.

They utilized intensity matching and height matching from one LIDAR sensor and multiple LIDAR maps. After that, an entropy-based optimization framework is used to sum up two separate matching scores for the importance of the map. Although this method suggests two map matching scores in the decentralized manner, the use of the single sensor is vulnerable in system failure. In addition, each map matching is not entirely heterogeneous because it is extracted from the same sensor. From this limitation, our motivation is that two heterogeneous map matching sources with two separate sensors and two maps can ensure optimal separation.

In this paper, we present a decentralized decentralized localization framework using point cloud and road network map matching sources. Two heterogeneous map matching sources are structured to be performed in a complementary manner. While point cloud map matching is carried out with height features, the camera sensor, together with the road network, performs map matching based on road surface features. Through stochastic situation analysis, the system balances the importance of the heterogeneous matching results. We guarantee real-time processing and precise pose in-

formation, while compensating each weakness and ensuring the stability of a particular sensor failure.

The main contributions of this study can be summarized as follows:

- We propose a joint usage of two different matching method which can ensure optimal separation. The two map matching methods are designed to be complementary to each other by decentralizing sensing targets and matching characteristics.

- We propose the stochastic situation analysis module to balance the importance of heterogeneous map matchings and to scrutinize system stability. The module conducts a comprehensive analysis of local maps and sensor attributes to obtain reliability weight with Dempster-Shafer Theory (DST).

The remainder of this paper is organized as follows. Section II describes the decentralized map localization framework while section III, IV, V and VI show details of proposed modules. Section VII shows evaluates performances of the proposed algorithm with experiments. Lastly, Section VII draws conclusions.

2.2 Framework

2.2.1 Sensor Configuration

System includes environmental sensors including multiple LIDARs and monocular camera, and motion sensors for in-vehicle sensors, IMU and low cost GPS. Low lost GPS is only utilized for approximating initial pose. In terms of maps used, LIDAR matching uses a height map, and camera-based matching utilizes a road network map. Monocular camera's distance value is approximated estimated by camera-LIDAR extrinsic calibration. When multiple LIDARs are used, LIDAR-LIDAR calibration is additionally performed. All sensory data is set to the vehicle center position. Sensor calibration is assumed to be accurate because it is not our main contribution.

2.2.2 Localization Framework

Localization system is designed to separately perform heterogeneous map matching sources and complementary integrated them afterwards. Fig. 2.2 illustrates utilized sensor and corresponding matching sources and how those are utilized within each function. The entire localization system utilizes the conventional monte-carlo particle filter model.

More specifically, overall steps of algorithm is shown in Algorithm 1. The pose candidates $\mathbf{x}_i$ are generated by in-vehicle/IMU sensors. For each candidate, extracted lanes and road surface markings from camera are matched to the observed lane targets from pre-built road network, while LIDAR's point features are matched with LIDAR height map. After acquiring reliability quantification scores (Q_{map}, Q_{sensor}) individually from local map capability and sensor capability aspect, the stochastic situational analysis computes reliability weights for heterogeneous matching sensors. The vehicle pose is estimated by selecting a pose candidate with the best map-matching scores, taking into account the reliability weights (γ, β) in matching importance analysis. After checking the system failure using *matching consistency analysis*, the integrated matching weight is calculated to get optimal pose $\mathbf{x}_{optimal}$.

2.3 Preprocessing for Road Network Map Matching

Thanks to the lane-level features in road network map, the matching provides stability on a wide road or high-traffic area where LIDAR system has difficulty capturing static objects around an ego-vehicle. To maximize road network matching, we propose methods to collect the road network level features from the camera and selectively take information from the map.

Algorithm 1: Multi-modal Localization using Heterogeneous Map Matching Sources

Input : Camera sensing measurement z_{camera},

LIDAR sensing measurement z_{LIDAR},

Camera local map m_{camera},

LIDAR local map m_{LIDAR},

A pose candidate list, $\{\mathbf{x}^i, w^i_{Integ}\}^N_{i=1}$

Output: Final pose, $\mathbf{x}_{optimal}$

$\{w^i_{LIDAR}\}^N_{i=1} \leftarrow Lidar\ Map\ Matching(z_{LIDAR}, m_{LIDAR}, \{\mathbf{x}^i\}^N_{i=1})$

$\{w^i_{Camera}\}^N_{i=1} \leftarrow Camera\ Map\ Matching(z_{camera}, m_{camera}, \{\mathbf{x}^i\}^N_{i=1})$

Funct. `Probabilistic Matching Importance Analysis(`$z_{LIDAR}, m_{LIDAR}, z_{camera}, m_{camera}$`):`

  $Q_{map}(g_{RN}, g_L) \leftarrow Map\ Quantization(m_{LIDAR}, m_{camera})$

  $Q_{sensor}(k_{RN}, k_L) \leftarrow Sensor\ Quantization(z_{LIDAR}, z_{camera})$

  $(\gamma, \beta) \leftarrow Matching\ Reliability\ Weight(Q_{map}, Q_{sensor})$

return (γ, β)

if $Matching\ Consistency\ Analysis(\boldsymbol{w}^N_{LIDAR}, \boldsymbol{w}^N_{camera}, \boldsymbol{x}^N) > thres$ **then**

  $\mathbf{w}^N_{Integ} \leftarrow (\mathbf{w}^N_{LIDAR})^\gamma \times (\mathbf{w}^N_{Camera})^\beta$

end

else

  if $\gamma > \beta$ **then**

    $\mathbf{w}^N_{Integ} \leftarrow \mathbf{w}^N_{LIDAR}$

  end

  else

    $\mathbf{w}^N_{Integ} \leftarrow \mathbf{w}^N_{Camera}$

  end

end

$\mathbf{x}_{optimal} \leftarrow argMax(\mathbf{x}^N, \mathbf{w}^N_{Integ})$

2.3.1 Lane Detection

Utilizing the lane detection module, multiple lanes have been detected. We leverage the previously proposed clustering algorithm for lane marking, Directional-DBSCAN[21]. This clusters in each lane to account for the direction of the cluster. The lane model, $y = C_0 + C_1 x + C_2 x^2$, is represented by the second order polynomial equation which gives lane level lateral offsets C_0, heading angle C_1, and curvature of the lane C_2 [20].

2.3.2 Selective Reading of Road Network Map

Our road network map is processed from raw 3D HD map provided by NGII (National Geographic Information Institute) of Korea. The HD map is drawn by a person based on a point cloud map, which is precisely produced through a mobile mapping system (MMS). The connection relationship among each other features is structured by id of link, which is the center of the lanes. For example, each lane has attribute information which is the id of the link on the left and the id of the link on the right. ID also has information on the index number of lanes and information about the color and shape of the lane. Similarly, the road mark also has the corresponding link's ID. Therefore, the closest link is searched to read directly left and right lane, and road marks when a current vehicle pose is provided.

2.4 Heterogeneous Map Matchings

This section introduces each of our designed heterogeneous map matchings. The area and number of pose candidates are a design-able factor that is a trade-off between computational cost and estimated pose accuracy. As the localized system is integrated, usually due to the cost of computing, the number and size of candidates decreases. The two matching measures share the same candidates. Each candidate state $\mathbf{x}_k^i$ $(1 \leq i \leq N)$ at time k is created to have an uniform pose interval. N and i indicate the number of candidate and an index, respectively. The shared state $\mathbf{x}_k^i$ for both matching

methods consist of 2D ego-vehicle global location (x_k^i, y_k^i) and heading angle (φ_k^i).

$$\mathbf{x}_k^i = [\ x_k^i \quad y_k^i \quad \varphi_k^i\ ]^T. \tag{2.1}$$

Our localization model also estimates the roll, pitch, and z values for use as prior knowledge of LIDAR's height map matching. Nevertheless, the main interest in integrating map matching data is modeled as XYW.

2.4.1 Road Network Map Matching

In road network map matching, road surface information such as lanes, road markings is used. Since the information obtained from the camera is local to the vehicle, the road network map to be compared is also brought locally based on the vehicle pose candidates.

The matching measurements $w_{CAM,k}^i$ derives using inputs of pose state $\mathbf{x}_k^i$, map m_k and sensing results z_k, where i presents i-th particle at k time. In terms of other road markings, local location vector of the road markings measured $\mathbf{m}_{RM}$ from the camera are compared to the generated road markings location vectors $\mathbf{m}_k^i$ on the map as viewed from the vehicle candidate poses.

$$w_{camera,k}^i = \exp - \left\{ \frac{1}{2} (\mathbf{m}_k^i - \mathbf{m}_{RM})^T \Sigma_{RM}^{-1} (\mathbf{m}_k^i - \mathbf{m}_{RM}) \right\}, \tag{2.2}$$

where $\mathbf{m}_{RM} = [x_{RM}, y_{RM}]^T$ is sensing measured 2D matrix and Σ_{RM} is a 2×2 covariance matrix.

To be more specific, we show a specific model when lane marking measurement is given. The lateral position of the ego-vehicle with respect to the driving lane $(C_{0,LANE})$ is calculated and compared with that of the generated pose (d_k^i), while heading angle $(C_{1,LANE})$ is calculated and compared with that of the generated angles (θ_k^i).

$$w_{Lane,k}^i = \exp - \left\{ (d_k^i - C_{0,Lane})^2 / 2\sigma_d^2 + (\theta_k^i - C_{1,Lane})^2 / 2\sigma_\theta^2 \right\} \tag{2.3}$$

where d_k^i and θ_k^i are calculated with respect to the lane in the road network map that includes the generated poses $(\mathbf{x}_k^i)$.

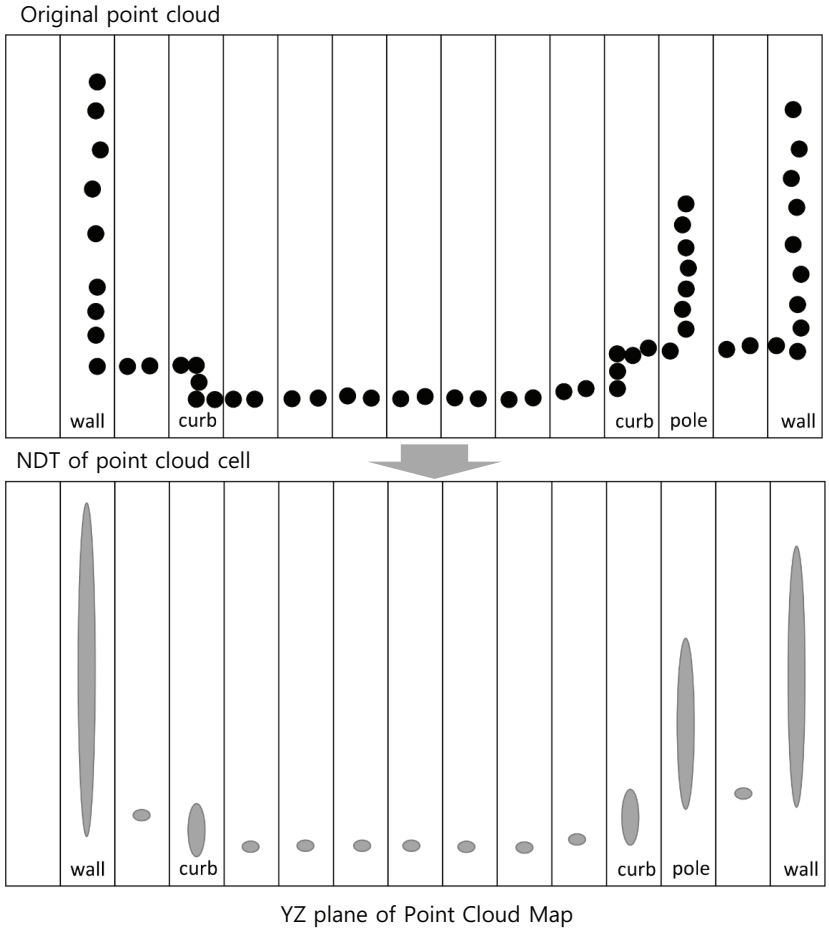


Figure 2.3. NDT of point cloud

2.4.2 Point-cloud Map Matching

This section briefly describes the utilized 3D matching method [52] since our contribution does not include enhancement of point cloud map matching.

A Point Extraction

Point cloud map matching method utilizes point extraction to the point cloud to reduce the number of points compared to the prior 3D map. Point clouds input is transformed to an NDT (Normal Distribution Transformation) representation by accumulating x, y,

and z values to a regular XY grid with a given resolution and distinguish ground points that were useful for RP (roll and pitch) search such as sidewalk and roads and vertical points that were useful for XYW search such as buildings, trees, and curbs.

The corresponding LIDAR spatial map is also generated using a point cloud map which is precisely produced using a mobile mapping system (MMS). Utilized features from the LIDAR spatial map also follows transformed NDT representation in order to properly corresponds to the sensing point cloud features. The transformed NDT features are pre-stored to reduce processing time.

Height Map Matching

Point extraction result is transformed from local frame to global frame, i.e. pose candidates, and compare point distribution between point extraction result and a prior height map. A prior height map has distribution of elevation of 3D point cloud map. The map's predefined feature format is similar to the sensor perspective, it is following NDT formation. By calculating difference between the point and map, we measures precise current pose of vehicle. The LIDAR matching measurement is calculated as below:

$$w^{i}_{LIDAR,k} = \exp\left\{ -\sum_{c} \left(\frac{z_c - \mu(x_c, y_c)}{\sigma(x_c, y_c)}\right)^2 \right\}, \tag{2.4}$$

where z_c is sensing point's height while $\mu(x_c, y_c)$ and $\sigma(x_c, y_c)$ are mean and variance of map's c-th grid.

2.5 Probabilistic Matching Importance Analysis

Matching Importance Analysis evaluates the performance and stability of matching sources to maximize the effect of merging map-matching data. The advantages of the two matching methods are complementary to each other in terms of driving situations and environments. Even if a particular sensor falls out at a vulnerable moment, the

overall system will not be compromised if the weak sensor takes up a small portion of the measurement. For instance, lane matching is effective when there are few static objects to match in a 3D environment, such as heavy traffic or wide road. Meanwhile, 3D high-density LIDAR matching is useful for roads with no lanes or road signs. In addition, the mutual use of two separate matching methods can compensate for malfunctions in an inherent failure.

2.5.1 Status Quantification: Overview

Every map matching has its characteristics depending on its map and sensor characteristics. Therefore, a certain matching character can be robust at certain condition while the other matching characters are fragile at the exact same condition. This can be caused by either sensor measurement, corresponding local map, or both of them. That means that a map and a sensor in the current circumstances can individually effect map matching performance. Therefore both maps and sensors are separately evaluated to find more reliable map matching source in the moment. This step evaluates the map and sensor data based on their characters and quantifies the data sufficiency of sensors and maps for use as input values for reliability-matching weight calculations. If a particular sensor or map contains more sufficient information, the sensor or map with sufficient information has higher chance to gain better matching-reliability weight. All target sources' models are following this rule as shown below:

$$Q_{score}(x) = \exp(-\lambda_{target}\Re(K_{Thres} - \sum_{i \in I} f(x_i))). \tag{2.5}$$

This indicates that the more features from the source, the higher capability to match with accuracy. However, it has saturated region when the source's feature exceeds its sufficient feature threshold K_{Thres} to have proper matching. If it exceeds, the matching system will no longer improve and this is set as a saturation parameter. Each map and sensor is reflected its own character and they are analyzed by feature reliability

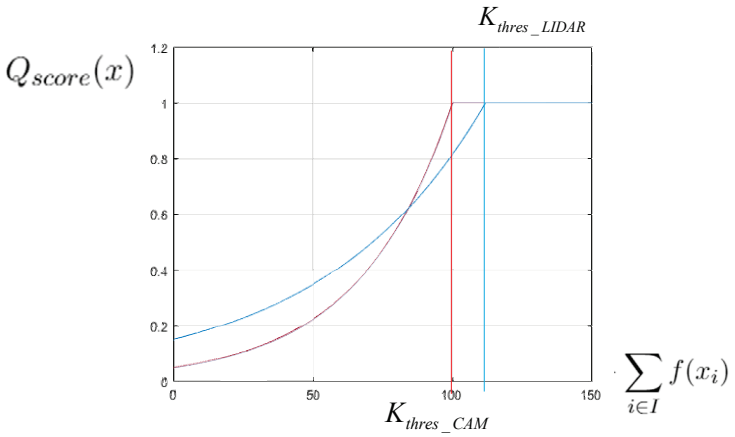


Figure 2.4. Suffciency score function graphs

quantification. Maps' quantified scores are $g_L(c_L), g_{RN}(c_{RN})$, while sensors' quantified scores are $k_L(x_L), k_{RN}(x_C)$.

2.5.2 Reliability Quantification of maps

The main principle of map quantification is to quantify sufficiency and up-to-date information of map data as a matching target.

Local map data sufficiency

Secondly, data sufficiency of current local map is key factor for map matching. Sufficiency of the map is based on regional characteristics and the characteristics of the map itself. In terms of regional characteristics, maps have very large scale data compared to sensing range. However, maps where map matching takes place are limited local region. We call it local map. Depending on the location, certain local maps may contain a wealth of information and some other local maps may not.

On the other hand, map originality varies with its original purpose. Sufficiency of road marking information of local map is important in road network, while sufficiency of static objects with inherent height feature in local 3D Point cloud map is key factor

in map matching.

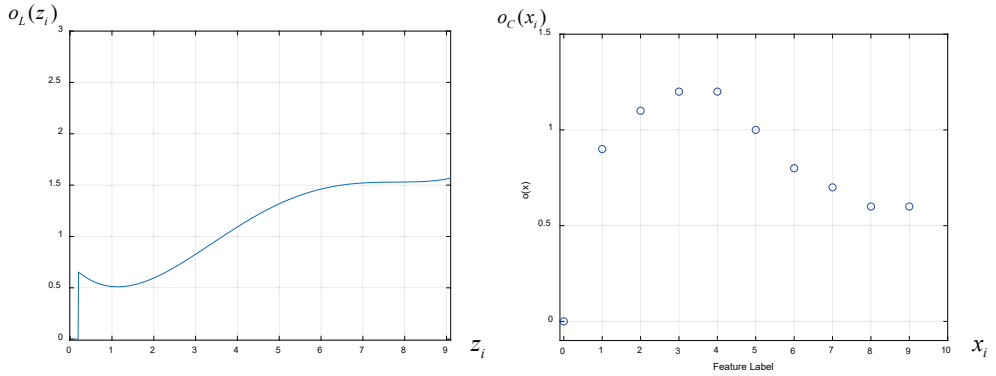
Local point cloud map data sufficiency

In terms of 3D point cloud map, richness of distinctive cell in the map is checked. Since variety of height feature in the sensor is important to match, a certain amount of distinctive cell with the large z variance or mean of point set is needed. If a road is broad and the adjacent area is flat, the cells in the local map have nothing to compare as seen in top left case in figure 2.6. On the other hand, narrow road with skyscrapers is described in top right in Figure 2.6. Regarding point cloud map, the quantified height map $g_L(c_L)$ is calculated as

$$g_L(c_L) = \exp(-\lambda_{L,m}\Re(C_{L,Suff} - c_L(X))), \tag{2.6}$$

where $c_L(X)$ presents a current LIDAR local map capability and the threshold $C_{L,Suff}$ indicates a substantial level of LIDAR local map capability. Hereby, $\Re(x)$ is formulated rectified linear unit function, $\Re(x) = \max(0, x)$. This is to apply the same maximum probability when the abundance of data is greater than a given threshold, which indicates sufficient enough.

The local map capability $c_L(X)$ is modelled concerning distinctive height value within the map grid. As shown in eq. 2.7, $c_L(X)$ is sum of each grid occupancy function $o_L(z(x_i))$ when given the i-th map grid x_i. A single grid occupancy function $o_L(z)$ is modelled concerning distinctive height value within the map grid. When the height of objects on a grid is clearly distinguished compared to ambiguous heights, such as pedestrian or sedan height, the matching ability is substantially meaningful. In this perspective, high frequent (z) we get, less distinctive this feature is. $o_L(z(x_i))$ is approximated in a quadratic polynomial model based on this LIDAR sensor measurement by setting the median frequent height (z) is set to one. The approximated function $o_L(z(x_i))$ is drawn in Fig. as shown in Fig.2.5-a). This shows that ambiguous object height between 1m and 2m has less confidence in feature occupancy factor. If



(a) Lidar occupancy function (b) Camera occupancy function

Figure 2.5. Occupancy function graphs

points with certain high height ($k_{high} = 13$), it has limit occupancy value because LIDAR point cloud coverage has limitation. Any points with less than ground level height ($w_{ground} = 0.4$) are filtered out since points nearby the ground can be outliers.

$$
\begin{aligned}
c_L(X) &= \sum_{x_i \in X} o_L(z(x_i)) \\
o_L(z(x_i)) &= \begin{cases} a_0 z^4 + a_1 z^3 + a_3 z^2 + a_4 z^1 + a_5, & w_{ground} \leq z(x_i) \leq k_{high} \\ o_L(z(x_i) = k_{high}), & z(x_i) > k_{high} \\ 0, & others \end{cases},
\end{aligned}
\tag{2.7}
$$

Local road network map data capability

In terms of road network map, richness of road marking information in the map is checked. Capability of road semantic information in local map, e.g. diversity of lane types, sufficient lane markings within the searching region, is an important singularity. Based on Baysian probabilistic theory, those factors are evaluated to derive reliability probabilities, $g_{RN}(c_{RN})$, on the quantified RN map as follows:

$$
\begin{aligned}
g_{RN}(c_{RN}) &= \exp(-\lambda_{RN,m} \Re(C_{RN,Suff} - c_{RN}(s, X))), \\
c_{RN}(s, X) &= f_d(s) \times f_S(X),
\end{aligned}
\tag{2.8}
$$

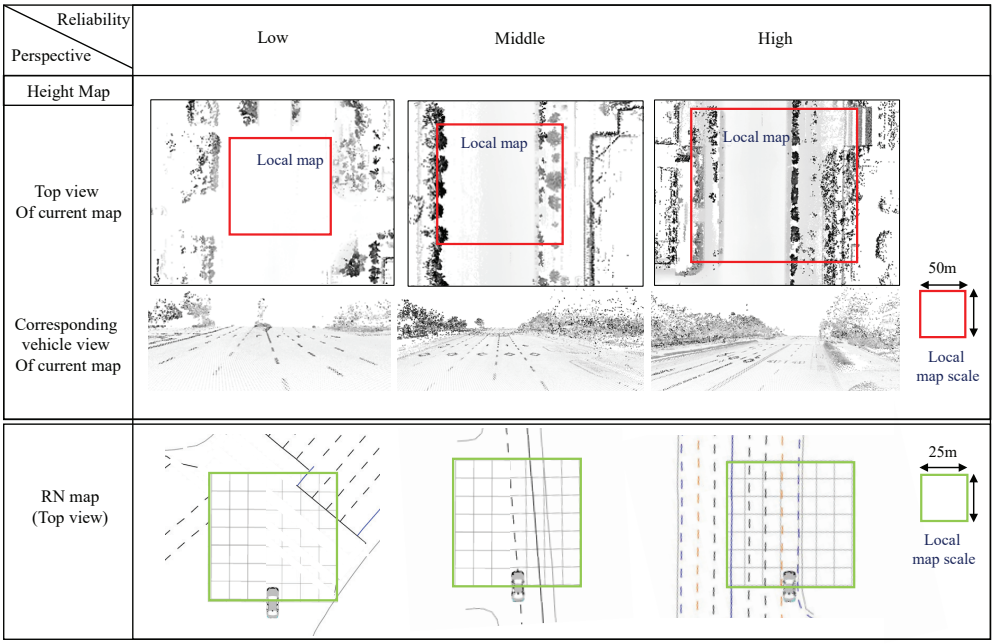


Figure 2.6. Examples of Map belief

where c_{RN} and $C_{F,Suff}$ indicates current capacity of the RN map and sufficient threshold of local RN map capability. Map capability function consists of semantic diversity factor $f_d(s)$ and local map sufficiency factor $f_S(X)$.

First of all, the map sufficiency function $f_S(X)$ presents the sum of feature sufficiency $f_c(x_i)$ in each grid, $x_i, i \in X$, where $n(X)$ is the number of searching grids. Each grid score $f_c(x_i)$, shown in Eq.2.17, is multiplication of ambiguity $v(x_i)$ and feature occupancy $o(x_i)$.

$$\begin{aligned} f_S(X) &= \sum_{x_i \in X} f_c(x_i) \\ f_c(x_i) &= o_c(x_i) \times v(x_i) \end{aligned} \tag{2.9}$$

All factors are designed within the grid basically. Among them, the main factor is feature occupancy $o_c(x)$. More grid with occupied feature we get the more score for this function. If the feature within the grid x_i has more distinctive feature such as yellow lane, more score can be acquired. Since the feature in this map and sensor has a

discrete label, the occupancy function would have discretized function accordingly as shown in Fig.2.5-b). If there are multiple features from different road networks within a single grid, it increases the ambiguity of map matching scores because it is confusing to compare. Considering that characteristics, $v(x_i)$ is designed as follows:

$$v(x_i) = \begin{cases} 1, & n(x_i) \leq 1 \\ -1, & others \end{cases}, \tag{2.10}$$

where $n(x_i)$ indicates the number of road network within the i-th grid.

Meanwhile, $f_d(s)$ is designed to consider semantic diversity of the road network map. Compared to map feature occupancy aspects in $f_S(X)$, $f_d(s)$ focuses on road network character itself. Given that existing lane number e, link number l and the number of different semantic label d, the function is designed as follows:

$$\begin{aligned} &s = d \times e/(l+1), \\ &f_d(s) = c\tau^{\delta_{\text{int}}}\gamma^{1-\delta_{\text{int}}}, \quad \delta_{\text{int}} = \begin{cases} 1, & s \geq 1.2 \\ 0, & others \end{cases}, \end{aligned} \tag{2.11}$$

This function verifies that there are sufficient number of lanes relative to the the number of lane link.

2.5.3 Reliability Quantification of Sensors

Sensor reliability quantification is performed to evaluate capability and accuracy of sensor measurement data as a matching source.Since map quantification is evaluated in consideration of sensor aspects, the modelling of each sensor quantification starts from a similar concept.

LIDAR sensor perspective

Similar to the LIDAR map perspective, height features from the static objects such as trees, curb, building or traffic infrastructures in point clouds are important sources for map matching. However, invalid dynamic objects, such as pedestrians or cars which

can not correspond to the pre-built map, are existing in the sensor measurement. Not only that, the point sparsity in the crowded region also occurs in such following situation: when the car is in high traffic and is surrounded by huge dynamic objects such as bus or truck, most scanned laser points are projected onto the object. The shape of the point cloud from bus can be confused with the building and there are few points that penetrate the map object with substantial distance. In this hard condition, point cloud matching source is in hard condition and can be less reliable.

In short, sufficiency of static object points with various distance are important matter to have meaningful map matching. To gather static object point cloud, scanned point clouds is processed under point cloud segmentation [59] for clustering points within the object-level after extracting ground point[60]. Among non-ground points, those are divided into static objects and dynamic object points. This would be solved by either point clouds' object classification[61] or predefined map based filtering [62]. First, point cloud classification is categorized by feature-based object classification, or ray tracing by tracking of dynamic objects. In terms of map based filtering, objects in the drive-able road in the road network map are filtered out as dynamic objects. After using either method, the classified static object points are rearranged into occupancy grid. The size of the grid is the same as the cell size of the 2D height map. After classifying the static points cluster, object points are placed to the grid cell. In order to follow the quantification method, the precision of the static object classification is pre-defined after evaluation of the algorithm. The precision is defined as:

$$precision(\kappa_L) = \frac{\#\text{ of correctly detected static objects}}{\#\text{ of whole detected objects}} \tag{2.12}$$

The LIDAR sensing capability quantification $k_L(x_L)$ is derived by

$$\begin{aligned} k_L(x_L) &= \exp(-\lambda_{L,z}\Re(X_{L,Suff} - x_L(X))), \\ x_L(X) &= \sum_{x_i \in X} f_L(z(x_i)), \end{aligned} \tag{2.13}$$

where $x_L(X)$ and $X_{L,Suff}$ indicates current capability of the a LIDAR sensor and sufficient threshold of LIDAR sensing capability, respectively. The LIDAR sensing

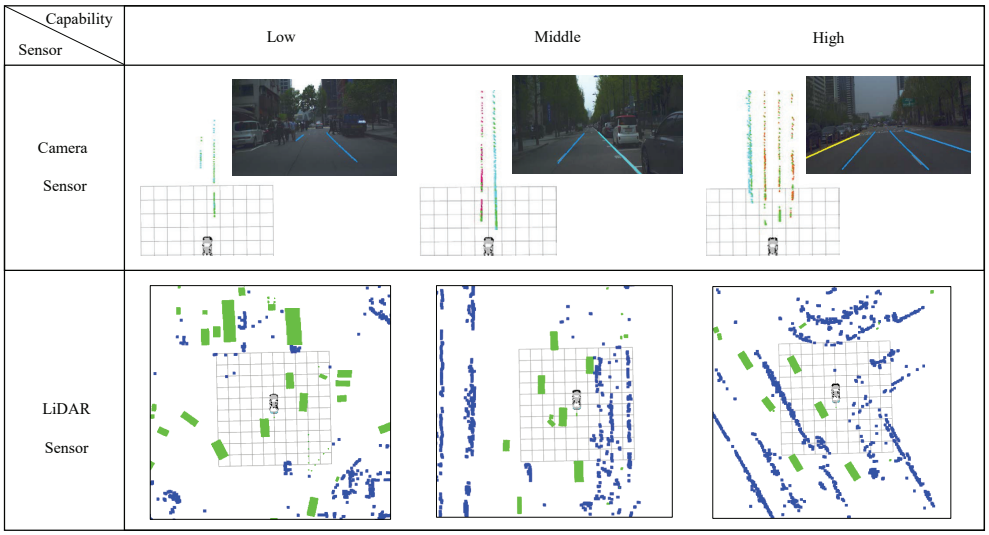


Figure 2.7. Examples of sensor belief

capability $x_L(X)$ is sum of the occupancy degrees of each cell $f_L(z(x_i))$ with height when x_i is a grid subset of all searching grid set X and $z(x_i)$ shows the height value of the i-th grid.

The occupancy grid function of LIDAR sensor data sufficiency is similar to the LIDAR map model. But the $o_L(z)$ from Eq. 2.7 is utilized concerning sensing factor. Static object classification is not ideal and precision is pre-defined as κ_L. The sum of classified static object with ratio κ_L and classified dynamic object with ratio $1 - \kappa_L$ are probabilistic static object as follows:

$$f_L(z(x_i)) = \kappa_L \cdot o_L(z(x_{i,\,Static})) + (1 - \kappa_L) \cdot o_L(z(x_{i,\,Moving})), \tag{2.14}$$

The second row in Fig. 2.7 illustrates that static objects are blue and moving objects are green. All of them are assigned to the grid cell $x_i,\ i \in \{1, .., N_{cell}\}$.

Camera Sensor Perspective

Sensing capability of detected lane and road marker from a camera sensor is quantified. Regarding camera sensor capability, the quantified camera sensor $k_{RN}(x_C)$ is

modelled as

$$k_{RN}(x_C) = \exp(-\lambda_{RN,z}\Re(X_{C,Suff} - x_C(I))), \tag{2.15}$$

where $x_C(I)$ and $X_{I,Suff}$ indicates current capability of the a camera sensor and sufficient threshold of camera sensing capability, respectively. Sensing capability function $x_C(I)$ consists of detection reliability factor $r_c(s)$ and corresponding feature sufficiency factor $S_c(X_i)$.

$$x_C(I) = \sum_{i \in I} r_c(i) \times S_c(X_i), \tag{2.16}$$

where I is a set of lane instances while X_i is a set of sensing grid involved i-th lane instance. First, in terms of detection reliability $r_c(i)$, lane and road marker detection algorithms are influenced by two processes: feature extraction and feature fitting. These performances are modelled as reliability function $r_c(i)$, which is the product of feature extraction's accuracy $c_f(i)$ and lane polynomial fitting accuracy $c_e(i)$. That is, $r_c(i) = c_f(i) \cdot c_e(i)$. Those accuracy and polynomial error are given information that accompanies detection algorithm.

On the other hand, feature sufficiency of i-th instance $S_c(X_i)$ is calculated as follows:

$$\begin{aligned} S_c(X_i) &= \sum_{x_j \in X_i} s_c(x_j) \\ s_c(x_j) &= \kappa_C \times o_c(x_j) \times m(x_j), \end{aligned} \tag{2.17}$$

where x_j is the j-th feature subset of i-th instance. Feature's sufficiency is a multiplication of κ_C, $o_c(x_j)$ and $m(x_j)$. Occupancy model $o_c(x_j)$ follows the same model as in RN map quantification. This model includes diversity of lane types and sufficiency of lane features as like in map evaluation, which enables a substantial sensing performance in map matching. κ_C indicates precision on marking extraction performance since $1 - \kappa_C$ portion of the detected lane features are false detection. Lastly, measurement reliability $m(x_j)$ presents the distance depreciation factor in camera sensors. It is difficult to guarantee distance accuracy in camera measurements inherently. The error majorly caused by transformation from a front view image to the top view coordinate.

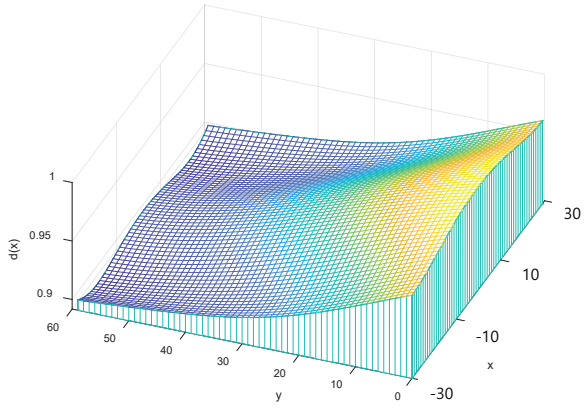


Figure 2.8. Measurement reliability function $m(x_j)$

Radial undistortion from lens and homography transform error for distance estimation are included. The distance function is modelled as shown in Fig.2.8 and Eq. 2.18.

$$m(x_j) = \sum_{c=1}^{5} \alpha_K N(x_j; \mu_c, \sum_c) \tag{2.18}$$

The parameters α_k, μ_c ,$\sum_c$ of gaussian mixture model(GMM) is determined from the distance variance observation of the single pixel representation. Those values are pre-trained value and describe futher in appendix.

2.5.4 Matching Reliability-Weight

Probabilistic situation analysis evaluates map and sensing status to obtain matching reliability weight. There are two steps: map and sensing reliability quantification and stochastic weight calculation. In reliability quantification step, map and sensing status are numerically quantified to find balance between two map matching sources. Combining each map and sensing reliability factors, matching-reliability weights for each map-matching are achieved based on Dempster-Shafer theory (DST).

Demster-shafer Theory

DST framework models a belief of an evidence with uncertainty, which has shown an excellent performance in sensor fusion research. Let X be a universal set that includes all states of interest. In the case of two propositions being considered, e.g. $X = \{a, b\}$, the power set is $2^X = \{\varphi, a, b, \{a, b\}\}$, where the degree of the belief, i.e. the mass, of each hypothesis is represented by a probability value in a DST framework. This is called Basic Belief Assignment (BBA), shown as $m : 2^X \rightarrow [0, 1]$ which has the following two characteristics:

$$m(\varphi) = 0 \tag{2.19}$$

$$\sum_{A \subseteq X} m(A) = 1 \tag{2.20}$$

When beliefs are obtained from multiple sources, the degrees of the common beliefs can be derived by Dempster's rule of combination as follows:

$$\begin{aligned} &m_{S,M}(\varphi) = 0, \\ &m_{S,M}(A) = (m_S \oplus m_M)(A) \end{aligned} \tag{2.21}$$

As the goal of this study is to find the importance of map matching sources, a universal set of two propositions, $X = \{L, RN\}$, is defined where *L* and *RN* denote a LIDAR map matching and a road network map matching, respectively. There are two beliefs for deriving the importance of map matching: local map sufficiency belief and sensor reliability belief. First of all, masses $m_M(A)$ for map reliability beliefs are designed as follows:

$$m_M(A) = \begin{cases} m_M(\{L\}) = (1 - \alpha_M) f_{Map}(c) \\ m_M(\{RN\}) = (1 - \alpha_M)(1 - f_{Map}(c)) \\ m_M(\{RN, L\}) = \alpha_M, \end{cases} \tag{2.22}$$

where $\alpha_M \in [0, 1]$ is an evidence discounting factor as in [85] to represent a degree of a belief that cannot be assigned to both {L} and {RN}.

$$f_{Map}(c) = \frac{g_L(c_L)}{g_L(c_L) + g_{RN}(c_{RN})} \tag{2.23}$$

In the case of sensor reliability beliefs, masses $m_S(A)$ are designed as follows:

$$m_S(A) = \begin{cases} m_S(\{L\}) = (1-\beta_S) f_{Sensor}(x) \\ m_S(\{RN\}) = (1-\beta_S)(1-f_{Sensor}(x)) \\ m_S(\{RN, L\}) = \beta_S, \end{cases} \tag{2.24}$$

$$f_{Sensor}(x) = \frac{k_L(x_L)}{k_L(x_L) + k_{RN}(x_C)} \tag{2.25}$$

From each map and sensor mass equation, The two mass sets above are combined according to weight is calculated as follows:

$$\begin{aligned} m_{S,M}(L) &= (m_S \oplus m_M)(L) = \frac{\sum_{B \cap C = L} m_S(B) m_M(C)}{1-K} \\ m_{S,M}(RN) &= (m_S \oplus m_M)(RN) = \frac{\sum_{B \cap C = RN} m_S(B) m_M(C)}{1-K} \end{aligned} \tag{2.26}$$

$$K = \sum_{B \cap C = \Phi} m_S(B) m_M(C) \tag{2.27}$$

When the joint mass is decided, the joint mass of lidar $m_{S,M}(L)$ is greater than the joint mass of road network $m_{S,M}(RN)$, the reliability of LIDAR map matching is higher than the road network map matching. In this moment, if fail-safe module tells discrepancies in matching consistency analysis, the more reliable matching takes responsibility for estimating pose. Otherwise, the both matching system is rather stable, the reliability matching-weights (γ, β) is utilized for balancing each matching sources and it is derived as the normalized mass value as shown in equation (2.28).

$$\gamma = \frac{m_{S,M}(\{L\})}{m_{S,M}(\{RN\}) + m_{S,M}(\{L\})}, \quad \beta = \frac{m_{S,M}(\{RN\})}{m_{S,M}(\{RN\}) + m_{S,M}(\{L\})}, \tag{2.28}$$

The sum of γ and β equals to one, i.e $\gamma + \beta = 1$.

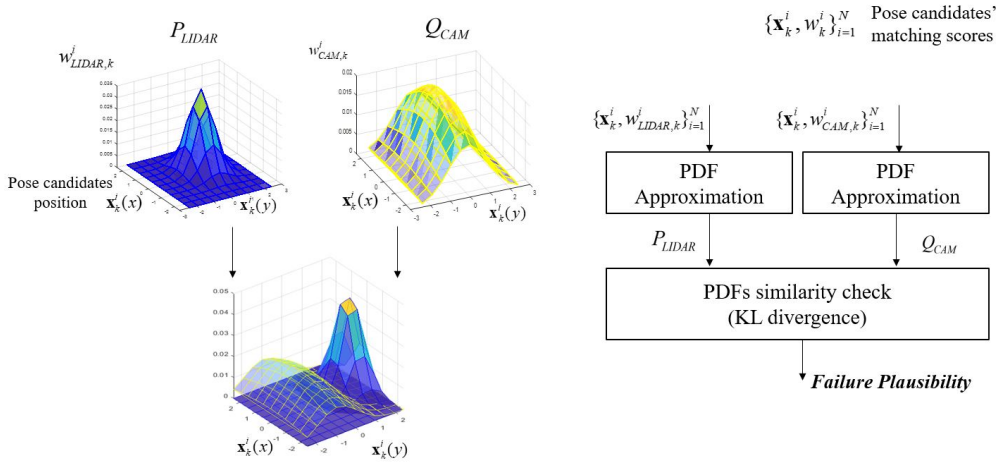


Figure 2.9. Dissimilarity detection of multiple matching sources.

2.6 Fail-safe Status Analysis

Probabilistic fail-safe status analysis evaluates suitability of current map-matching sources. Via this analysis, the system detects the failure moment of overall system or a certain map-matching source and making other matching source to compensate for the failure. Not only that, the size and area of pose candidates $\mathbf{x}_i^n$, can also vary according to the stability of the system. This section solves two problems. First, it analyzes the stability of the system by observing dissimilarity of two matching sources. After that, the system detects perfect failure when the dissimilarity persists.

2.6.1 Matching Consistency Analysis

The stability of the localization system is based on the similarity of two matching sources. The monte-carlo based localization system manages a group of pose candidates for pose estimation. The pose candidate group contains a set of pose $\mathbf{x}(x, y, yaw)^N$ and its matching weights $\mathbf{w}^N_{LIDAR}$, $\mathbf{w}^N_{cam}$. Each pose $\mathbf{x}_i$ has two matching scores $w^i_{LIDAR,k}$, $w^i_{cam,k}$ as a result of individual map matching procedure.

A probabilistic modelling is adapted to model score function of map matching

weights. Each matching result can be approximated as a multivariate probabilistic density function (pdf) with x, y and the corresponding weight value. We design the candidate groups as a Gaussian distribution. The probability distributions of the pose candidates are different for the matching source characteristics. $P(x, y)$ is the modelling of LIDAR map matching, whereas $Q(x, y)$ is the modelling of road network map matching.

$$D_{KL}(P_{LIDAR}||Q_{CAM}) \geq T_{Diverge} \tag{2.29}$$

Adapting the concept of the KL-Divergence, the difference between probability distributions P_{LIDAR} and Q_{CAM} is calculated. In our condition, the seed pdf is normally set to be more reliable matching source. As shown in Fig. 2.9-a), it would theoretically be the same position to have the measurement peaks if both matching results are perfect. This would mean that the system would behave well. Meanwhile, one matching result would be unreliable if two matching distribution is dissimilar as shown in Fig. 2.9-b). In the figure, the calculated difference between the two sources is greater than a certain threshold $T_{Diverge}$, which indicates minimum dissimilarity between two map matching sources.

2.6.2 System Instability Response

In aforementioned case, the converged pose candidates with shrank coverage region do not cover the actual pose region. To make it safe, the number and the area coverage of the pose candidates are increased since the prior scanning coverage cannot guarantee the stability any longer. If the unstable moment persist, a certain map matching source from the localization system is considered to be failed. A matching source with a bigger reliability-weight takes the initiative in matching score results since the analyzed reliability mass tells which measure is more stable.

2.7 Localization Model

2.7.1 Motion Update

The state of next time step is estimated based on the system model. A vehicle kinematic model with on-board sensor inputs is used for time updates as shown in eq. 3.12.

$$\hat{\mathbf{x}}^i_{k+1} = \begin{bmatrix} \hat{x}^i_{k+1} \\ \hat{y}^i_{k+1} \\ \hat{\varphi}^i_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{x}^i_k + \tilde{\nu}\cos(\hat{\varphi}^i_k + \tilde{\omega}\Delta t) \\ \hat{y}^i_k + \tilde{\nu}\sin(\hat{\varphi}^i_k + \tilde{\omega}\Delta t) \\ \hat{\varphi}^i_k + \tilde{\omega}\Delta t \end{bmatrix} \tag{2.30}$$

The control input of the vehicle model can be obtained from the vehicle motion sensors: a yaw rate sensor w_{gyro} and wheel speed sensors V_{whl}. We can predict the next step's position candidates where ϵ_v and ϵ_g are gaussian random variables. The shape of the probabilistic distribution can be changed via the configuration of sensor noises. $\tilde{\nu} = V_{whl} + \epsilon_v, \quad \tilde{\omega} = \omega_{gyro} + \epsilon_g$

2.7.2 Integrated Measurement Update

For integrated measurement update, each weight is updated by evaluating the likelihood of map matching measurements $p(z_k|\gamma, x^i_k, m_k)$ conditioned on reliability-matching weight γ, the predefined i-th state x^i_k, sensing measurement z_k and the local map map m_k at time stamp k. The relationship between the weight and likelihood can be expressed as $w^i_k \propto w^i_{k-1} \cdot p(z_k|\gamma, x^i_k, m_k)$. To simplify the calculation, the final weight of the particle w_k is determined as $w^i_k = w^i_{k-1} \cdot {w^i_{CAM,k}}^{1-\gamma} \cdot {w^i_{LIDAR,k}}^{\gamma}$, where reliability-matching weight (γ, β) balances each map matching's weight and $w^i_{LIDAR,k}$, $w^i_{CAM,k}$. If certain sensor failed, integrated score is reflected either $w^i_k = w^i_{k-1} \cdot w^i_{CAM,k}$, or $w^i_k = w^i_{k-1} \cdot w^i_{LIDAR,k}$, based on the more reliable sensor, derived from the matching source importance analysis.

Each map matching weight is calculated at the heterogeneous matching step. As for all calculated final weights $w^i_k (i \in \{1, .., N\})$, w^i_k is normalized by equation

$\hat{w}_k^i = w_k^i / \sum_{j=1}^{N} w_k^j$, where N is the number of particles. This sets the sum of all final measurement weights to 1.

2.7.3 State Estimation and Resampling

The final position of ego-vehicle is to be determined by maximum a-posteriori (MAP) estimation, which can be approximated as follows:

$$\hat{x}_k = \underset{\hat{x}_k^i}{argmax}\, p(x_k^i|z_{cam,k}, m_{cam,k})^{\beta} p(x_k^i|z_{LIDAR,k}, m_{LIDAR,k})^{\gamma} \tag{2.31}$$

Resampling regenerates the new set of uniformly generated candidates on the basis of their relative likelihoods. If no fault is detected during the probabilistic situation analysis phase, the resampling phase diffuses several concentrated candidates at the next step. If a dissimilarity or a fault is detected, the coverage region of the candidates to search increases for a larger area with uncertainty in the next step $k + 1$.

2.8 Experiments and Evaluation

2.8.1 Experimental Set-up

Sensors, Maps and Dataset

The proposed method was demonstrated through real environmental data collected from the survey autonomous vehicle shown in Fig. 2.13. The survey vehicle is equipped with a front camera (UI-3250CP-C-HQ) with FOV 87° lens, two Velodyne LiDAR (VLP-32), a RTK-GPS(OXTS RT3002), and a IMU (Xsens Mti-710). The IMU and in-vehicle odometer sensing results are exploited to predict the vehicle pose and two LIDAR sensors and front camera are utilized as measurement sensors. Although the high performance GPS has centimeter-level accuracy in RTK (RealTime Kinematic) mode, the RTK mode is intermittently available in the test areas due to skyscrapers. Therefore, the ground truth is refined poses utilizing a graph optimization method with

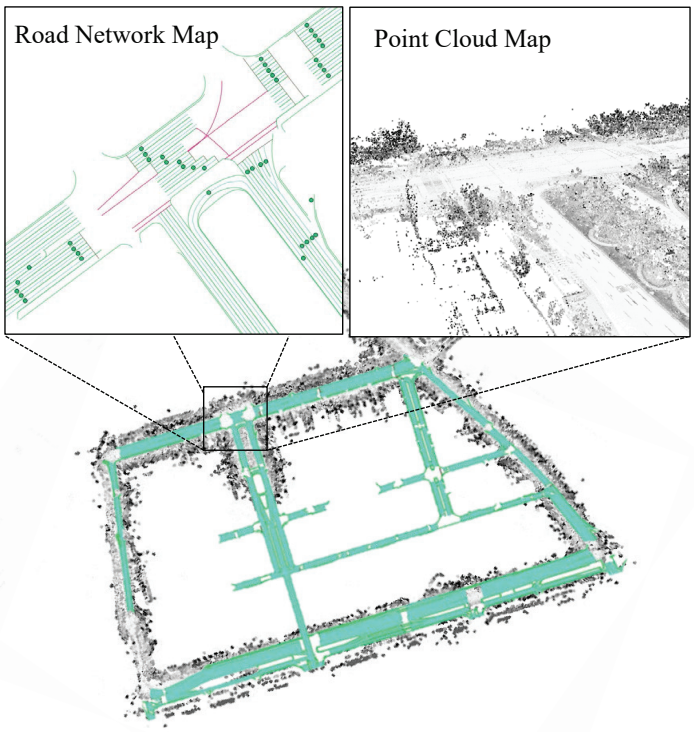


Figure 2.10. Two types of maps for trajectory B.

the high performance GPS, wheel odometry, and Iterative Closest Point (ICP) between measurements and the off-line map. After building spatial point cloud map, the road network map is made based on the spatial information and the coordinate system of the two map is set to be consistent. The road network and point cloud map are prepared in advance. The example map corresponding to trajectory B is shown in Fig. 2.12.

Two urban roads are tested for performance evaluation, Yeouido Seoul, Korea and Palo Alto, United States. First of all, Figure 2.11 shows a digital map depicting a road network map of the Yeouido and it shows two vehicle trajectories in red and cyan colors. We verify the effectiveness of introducing heterogeneous matching with experiments on the Yeouido dataset. Dataset A and B with different target zone are gathered to show various road conditions respectively. The target zone is considered as a legitimate test environment because the dataset includes a very wide road or a fairly narrow one-way road with sparse lane indications. In this dataset, we expect either lane marking based matching or LIDAR matching can be intermittently unstable based on their weaknesses. Secondly, Figure 2.11 shows a point cloud map in Palo Alto and vehicle trajectory is shown as yellow color. This area has a narrow bidirectional-way road with sparse lanes. Therefore it follows pattern in between zone A and zone B, but more similar to the zone B Yeouido dataset. Overall, A trajectory is 4.3 km path and B trajectory is 3.7 km path in Yeouido, while Palo Alto dataset contains 3.9 km path.

System and Parameter Setup

The localization algorithm is implemented using C++/OpenCV and operated using ROS to ensure real-time functionality. The algorithm runs at a frame rate of 15.4 fps on a normal PC (Intel Core i5 Q720 CPU @ 2.2 GHz, 12GB RAM and GeForce GTX 1080 Ti). Each step's computing cost results is displayed in Table 2.2. Since the two heterogeneous matching sources are processed in parallel, the total processing time is 66 ms. This meets the real-time processing needs of autonomous vehicle considering that LIDAR scanning costs 100 ms per one rotation.

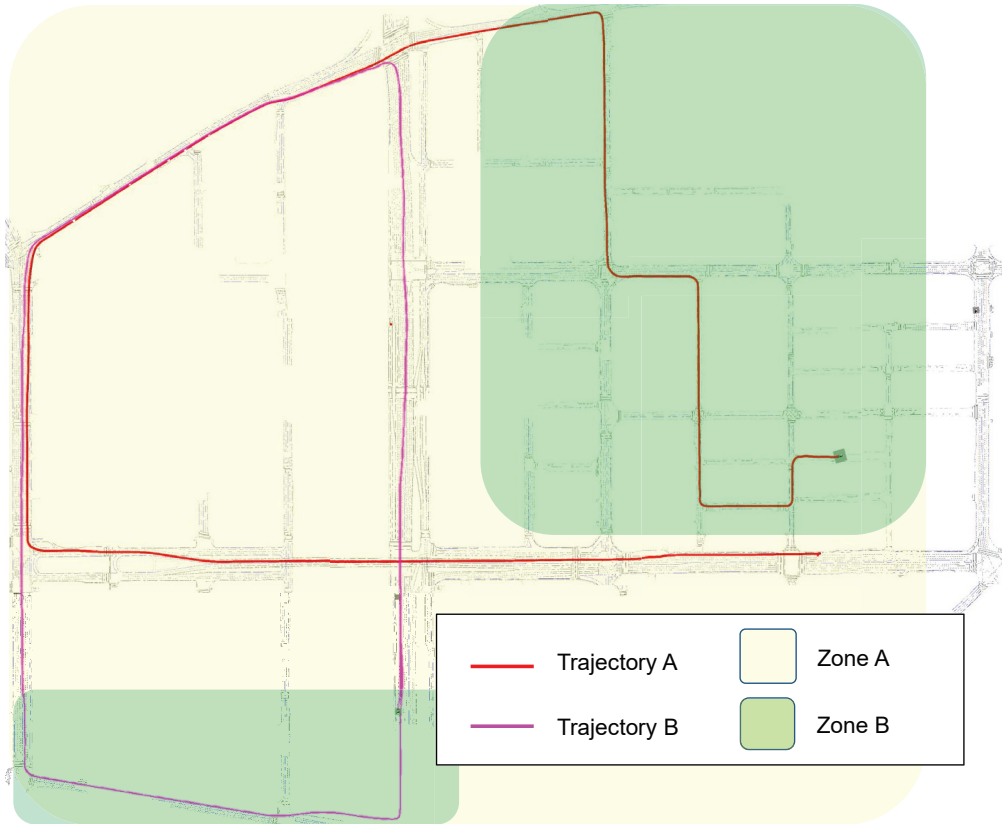


Figure 2.11. Localization results at Yeouido (shown in road network map).

Table 2.2. Computational cost at each step

Function	Processing time (ms)	
	Point cloud Matching	Semantic Matching
Motion prediction	3ms	
Matching (multi-thread)	49ms	43ms
Integration & motion update	16ms	
Total processing	68ms	

The parameters in the DST framework are set as follows, $\alpha_M = 0.18, \beta_S = 0.1$. The uncertainty parameter is larger in map since sensor's feature sufficiency and accuracy is more important than local map's. For quantification function parameters, each quantified sufficiency threshold values are set as $X_{L,Suff} = 150, X_{C,Suff} = 100, C_{L,Suff} = 170, C_{RN,Suff} = 90$, while saturation curve parameters are $\lambda_{L,z} = 0.02$, $\lambda_{RN,z} = 0.03$, $\lambda_{L,m} = 0.015$, $\lambda_{RN,m} = 0.025$. The designing rule follows assumption that the more gentle lowering factor is applied to empirically more reliable sensor matching.

2.8.2 Experiments & Evaluations

The contributions of this study can be highlighted through integrated map-matching performance comparison with single map-matching, and through system stability in the event of one sensor's failure, creating artificial errors.

Overall Performances

First of all, we evaluated the localization performance of LIDAR map-matching, road network map-matching and the proposed map-matching. Table 4.5 shows the localization performance when the test vehicle drives the target path as in Fig. 2.11. Although orientation is not effectively improved, most performance is advanced by integrated

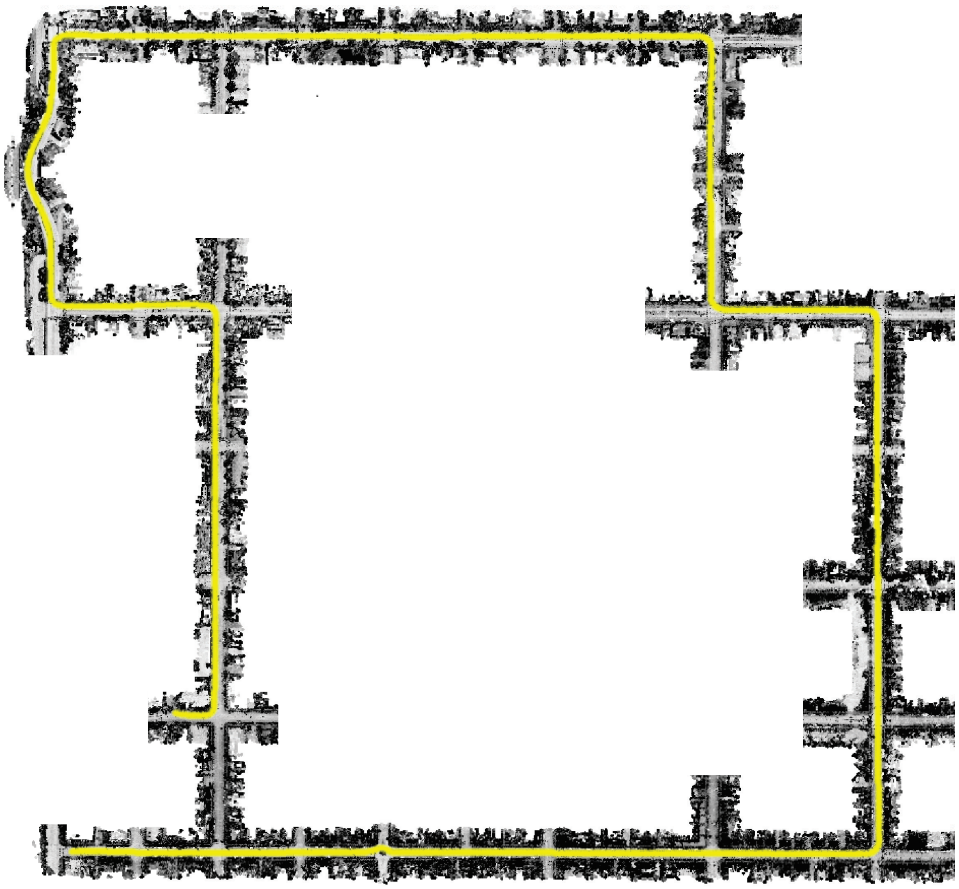


Figure 2.12. Localization results at Palo Alto (shown in point cloud map).

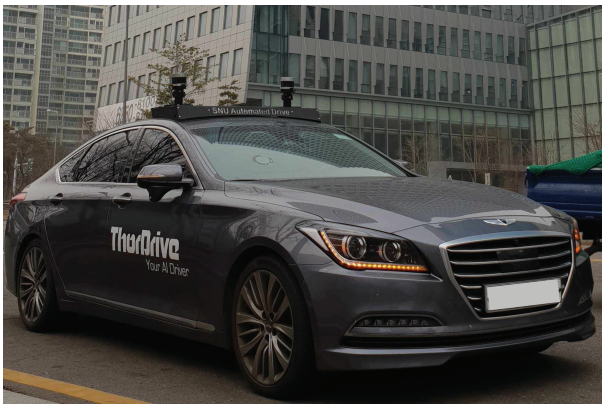


Figure 2.13. Autonomous vehicle test platform

Table 2.3. Localization Performances in Yeouido : Mean (Standard dev.)

Scenario	Zone A			Zone B		
Accuracy / Methods	Lateral(m)	Distance(m)	Orientation(°)	Lateral(m)	Distance(m)	Orientation(°)
LIDAR Map-matching [52]	0.396 (0.416)	0.634 (0.810)	**0.186** (0.139)	0.151 (0.147)	**0.244** (0.258)	**0.201** (0.172)
Road network Map-matching [28]	0.223 (0.301)	1.381 (1.58)	0.319 (0.296)	0.351 (0.408)	1.581 (1.776)	0.450 (0.383)
Integrated Map-matching	**0.156** (0.161)	**0.318** (0.392)	0.215 (0.284)	**0.149** (0.116)	0.283 (0.298)	0.208 (0.188)

map-matching. We divide the zone A and zone B in Yeouido data in order to see effects on each map matching's merit. Since zone A, with multiple lane conditions, has better results in camera matching, the integrated results can compensate error conditions in LIDAR matching. Zone B works vice-versa. Alley roads makes road marking map matching less effective and overall matching cannot be enhanced. On the other hand, condition in Palo Alto shown in Table 2.4 has similar to zone B, but has more lanes to match. This means that integrating heterogeneous methods can avoid the localization system from extreme failure. However, as a side effect, fusion values can sometimes result in decreased longitudinal accuracy because the lane map-matching to be merged includes longitudinal error, which acts as an inherent limit. However, integrated utilization is still meaningful since not only can it balance two models, but it also avoids sticking to big errors.

More specifically, Figure 2.14 demonstrates the advantages of the multi-modal map matching framework in depth. In the figure, the test vehicle is progressing from left to right and from zone A to zone D. The corresponding route is approaching an underground road, which is divided into two lanes of underground roads and four lanes of roads next to underground roads. In the map illustration, circles indicates the output trajectory of each localization result. The red trajectory represents the height based point cloud map matching results. Blue is the proposed map matching, while Magenta

Table 2.4. Localization Performances in Palo Alto : Mean (Standard dev.)

Methods \ Accuracy	Lateral(m)	Distance(m)	Orientation(°)
LIDAR Map-matching [52]	0.210 (0.201)	**0.341** (0.261)	0.195 (0.149)
Road network Map-matching [28]	**0.179** (0.204)	1.701 (1.806)	0.351 (0.273)
Integrated Map-matching	0.186 (0.156)	0.403 (0.439)	**0.191** (0.170)

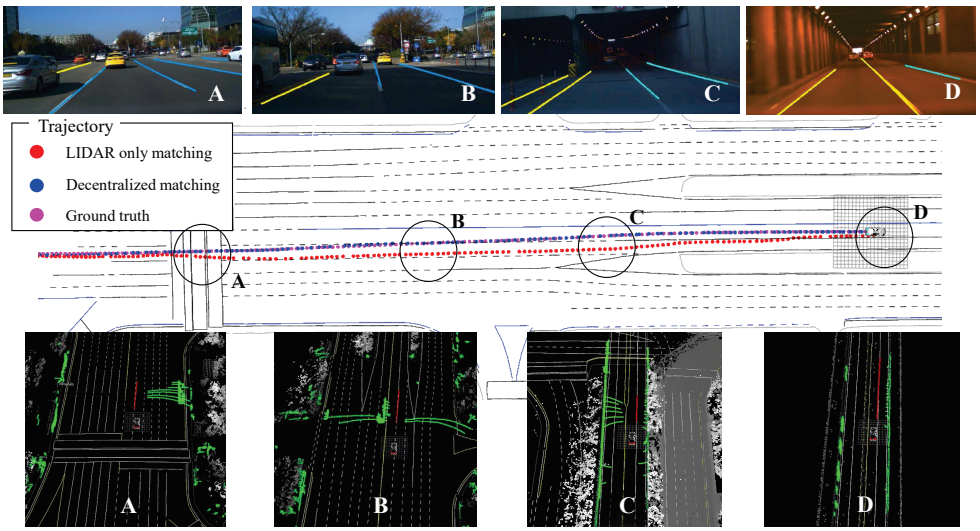


Figure 2.14. A scenario of localization result

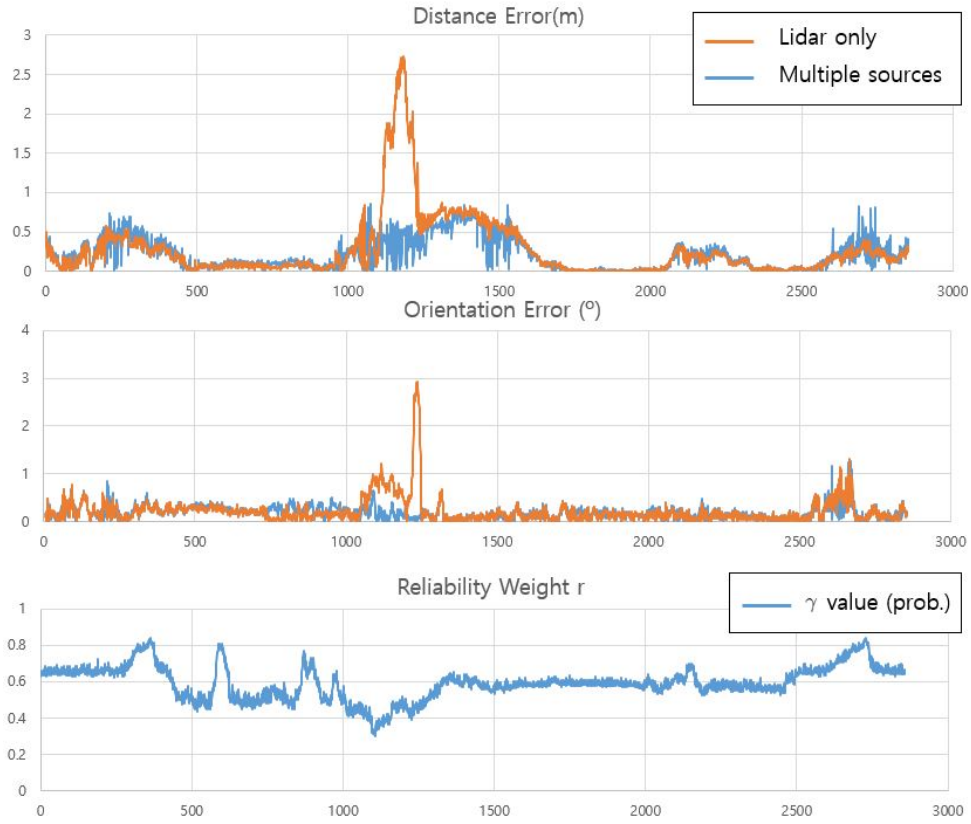


Figure 2.15. Localization Performance at a certain interval.

is ground true trajectories.

When entering zone A, the road is extremely broad containing 6 lanes in uni-direction, which is 12 lanes in bi-directional way. Note that the ego-vehicle is in the middle of a wide road, this section contains sparse scanned points and local height map with few traffic structures. This makes LIDAR only map matching fragile. Driving from zone A to zone B, the vehicle is changing lane from second lane to first lane. In C zone, the road is approaching to an underground road and the ego-vehicle is driving in the underground road in zone D.

In LIDAR map matching method, the point cloud measurement is considerably sparse and it depends highly on the vehicle dynamics of the IMU sensor. Unfortunately, in this test case, IMU information provides slightly contaminated data, which results in cumulative lateral errors. The pose error makes the vehicle misunderstand that it is still running in second lane and even improperly estimate the pose close to the wall in zone C, although it is obviously changed in first lane. This makes one-lane deviation between ground truth and estimated pose. After much more driving, the vehicle returns to its proper position by reaching the narrow road environment in zone D.

Meanwhile, the proposed map-matching schema works differently. In zone A, sparse scanned points and local map are quantified as low capability and this would result in falling LIDAR reliability weight γ under 0.4. Using our multi-modal concept, the lane based map matching part have more reliability weight scores and takes over the LIDAR map matching failures in between zone A and B. Because of the increase in reliability weight of camera map matching, the lateral offset maintains an accuracy of 10 cm even in lane change scenarios in very large areas and does not diverge unlike LIDAR only map matching. At the same time, there is a disadvantage that the inherent longitudinal inaccuracy of camera matching increases to a certain level as shown in figures in Fig. 2.14.

Matching reliability aspects

As an aspect of system robustness, weight variation and fail-safe situation have been evaluated in a qualitative way. First of all, reliability weight variations on various circumstances are shown according to the error of the LIDAR matching moments (frame 1100-1400). According to the quantification between the sensor measurements and local maps, the reliability (γ) is varying as shown in the third graph of Fig. 2.15. This caused by parameter tuning resulting from higher longitudinal accuracy in point cloud map matching in both stable condition. In the first fragile moment sequence at aforementioned Fig. 2.14, point cloud map matching is no longer reliable and it has the minimum reliability weight. However, at the second fragile moment (frame 2100-2300), the reliability weight gets higher since camera matching result is less reliable and LIDAR map matching compensates camera's fragile moment.

This matching-reliability weight is captured depending on the driving situation in Table 2.5 in order to elaborate the suitability of reliability weight. Only LIDAR reliability weight γ is displayed since camera reliability weight is $\beta = 1 - \gamma$. The matching importance of point cloud map matching is generally higher than the road network map matching: average of LIDAR reliability weight is set to be 0.57. However, the result of individual matching result performance and reliability weight are commensurate with each other. Camera matching reliability has more reliability compare to other sequences in the broad road since lane is sufficient in broad roads while LIDAR scanning points sparse compared to the narrow roads. Additionally, the closer to the center lane, the more reliability in camera map matching we got. In broad roads, intersection has lack of lane markings but at the same time, scanned points are also sparse. Therefore, it substantially balances. In narrow regions, point cloud map matching reliability increases. Especially, the proposed map matching in alley streets highly depends on the point cloud map matching. The results in the table shows that derived LIDAR reliability weight γ follows the actual matching ability in each map matching source under varying diverse circumstances.

Table 2.5. LIDAR Reliability Weight (γ): Mean (Standard dev.)

Scenarios	Each performance (m)	Camera Matching Reliable region	LIDAR Matching Reliable region
	LIDAR matching	0.719	0.137
	Camera matching	0.218	0.913
LIDAR Reliability Weight (γ)		0.479 (0.084)	0.641 (0.091)

Table 2.6. LIDAR Reliability Weight in different road scenarios (γ)

Scenarios / Target	Broad road				Narrow road	
	center lane	middle	last lane	intersection	alley	two-lane road
LIDAR Reliability Weight	0.469	0.510	0.599	0.567	0.696	0.581
Mean (Standard dev.)	(0.088)	(0.078)	(0.096)	(0.112)	(0.071)	(0.091)

Secondly, since it is difficult to capture multiple moments of matching failure, we have created a simulation in which errors are generated. The performance of multi-modal fusion-based matching and single matching was evaluated when the consistent error was inserted into one sensor's match. The LIDAR map was set 0.45 m away from the original pose, resulting in an artificial error. As shown in Fig. 2.16, advanced compensation for a particular sensor error was demonstrated. By complementing the multiple matching results, the position error is reduced to approximately 0.15 m.

2.9 Conclusion

This paper has proposed decentralized localization framework using multiple map matching sources including point cloud and road network map matching sources. This approach allows the system to be more robust on a certain sensor failure. Considering that the effective pose candidate parameters for each sensor matching source are different, the following study will cover these matching characteristics.

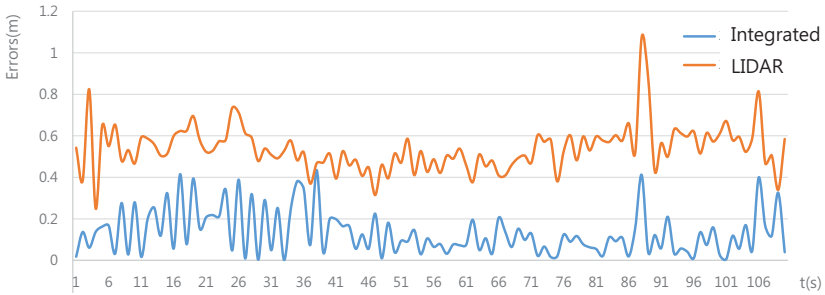


Figure 2.16. Localization performance comparison in artificial errors.

2.10 Appendix

In order to model the inevitable camera measurement errors, distance coverage within individual pixel is observed. By numerous observations from different terrains and driving pitches, observations of a single pixel's distance coverage variance $x_i \in \mathbf{x}$ $(0 \leq i \leq N)$ are collected. Based on the observation $\mathbf{x}$, image distance reliability is modelled by Gaussian mixture model (GMM). Assuming independence of observations x_i, we derive Gaussian parameters by a maximum likelihood function as:

$$(\hat{\mu}, \hat{\sum}) = \arg\max_{\mu,\sum} \sum_{i}^{N} \ln p(\mathbf{x}_i|\mu, \sum), \tag{2.32}$$

$$p(\mathbf{x}_i|\mu_k, \sum_k) = \frac{1}{(2\pi)^{D/2}|\sum_k|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_k)^T \sum_k^{-1}(\mathbf{x}-\mu_k)\right\}. \tag{2.33}$$

The parameters α_k, μ_c ,$\sum_c$ of GMM is determined value of approximated distance error rate by expectation and maximization algorithm.

Chapter 3

Ego-lane Index-aware Vehicular Localization using Deep-Road Network for Urban Environments

3.1 Introduction

Knowing the ego-vehicle's position, called vehicle localization, is indispensable for autonomous driving of vehicles and for the adaptive driver assistance system (ADAS). Accurate localization can help a navigation system provide more adequate routing instructions. For safety, localization provides semantic information for swerving to an adjacent lane to the ADAS system in order to to avoid an impending collision with another vehicle.

Among vehicle localization system, road marking based localization using road network map and road marking detection algorithm has been widely used due to its map compactness and its intuitive character of urban roads. This method is to find the ideal vehicle pose by calculating the matching score between detected road markings and local road network information from the map. Those methods are widely used methods. Due to road network's simple information, it is occasionally difficult to maintain localization accuracy compared to point cloud-based dense maps. For example, in mega-city, little measurements are made at large road intersections, or map

matching results may be somewhat ambiguous when a measured lane is aligned with an adjacent lane map from a number of lanes on the map.

In this paper, we propose an ego-lane index-aware vehicular localization system using multi-lane, ego-lane index and road marking information. For extracting road markings and ego-lane index, we also propose integrated DeepRoad network which extracts various types of road markings, lanes and ego-lane index within one single process. The proposed ego-lane index-aware localization algorithm is comprised of integrated DeepRoad network and map-matching algorithm using the extracted information from integrated DeepRoad Net. Main contributions of this study can be summarized into three folds as follows:

- We propose a new lane marking based localization system utilizing multiple visual features including multi-lane and ego-lane index. This provides better localization accuracy after the intersection region or broad regions. The approach provides the benefit of robustness in the broad roadways or urban scenarios. We confirm the performance of the proposed algorithm in diverse scenarios with spacious urban and highway cases up to six lanes, which have not been considered in previous studies.

- In order to put measurement to the proposed localization system, multi-lane detection and ego-lane index recognition results are needed. This is solved within one integrated marking extractor robustly by proposing integrated DeepRoad network. The integrated DeepRoad network recognizes lanes with lane types, road markings, and ego-lane information. Even in the case of insufficient visual contexts caused by hidden road information due to an object, the proposed approach can derive ego-lane index recognition significantly well.

An overview of the proposed framework is illustrated in Figure 1. The road information including road boundary and multi-lane information are exploited in the first network. This detection results work as an input for second network, i.e. ego-lane in-

dex identification network. The whole network is called integrated DeepRoad-net. Afterwards, using collected meaningful context road information, ego-lane index-aware localization system which is much stabilized method to only lane-based localization.

The remainder of the paper is organized as follows. In section II, related works of lane detection, ego-lane identification, and ego-lane index aware localization are introduced. The lane detection and ego-lane identification network are proposed in section III. and IV. Using those information, localization schema is proposed in section V. Demonstration and evaluation of each proposed algorithm are provided in section VI and conclusions are drawn in section VII.

3.2 Related Works

Conventional approaches for vehicle localization utilize some combined information from a high-precision Global Positioning System (GPS), an inertial measurement unit (IMU) and a digital map [1] [2]. With prior knowledge of the path, the known approaches show stabilized localization performance. These approaches, however, have several limitations on providing reliable and accurate localization results. Most of the typical GPS receivers provide rough information on the vehicle location with 2 to 25 meters deviations from the real position, which is not sufficient to attain the goal of this study. Especially in case of the urban driving, poor satellite geometry and disruption of radio signal reception due to skyscrapers further lower the accuracy of the lateral position. Compared to these approaches, other studies use visual sensors for a lane-level localization, on which lane the ego-vehicle is, that are relatively inexpensive to solve this problem.

Since the proposed localization method targets three step-wise aspects including lane detector, lane index inference, and index aware lane based map matching equation in this paper, this section handles those three aspects and how our contribution is differs from the previous works.

3.2.1 Lane Detector

Conventional lane detection methods (e.g. [8] [11] [20] [12] [17]) rely on a combination of handcrafted filters for feature extraction and heuristic association to identify lane segments. Popular methods of such hand-crafted cues include color-based features [20], ridge filter [11] and, cone-hat filter [22], which are possibly combined with a Hough transform [17], and Kalman filters [8] or particle filters [11] . After extracting the lane markings and segments, post-processing techniques (e.g. conditional random fields [12]) are employed to elicit mis-detection and group segments together to form the final lanes.

Normally, these traditional approaches are easy to have robustness issues because of environment changes that can not be easily modeled. More recent methods have replaced the hand-crafted feature detectors with deep neural networks to learn deeper and robust features, i.e. pixel-wise lane segmentation. Gopalan et al. [33] utilizes a pixel-hierarchy feature descriptor to model contextual feature and a boosting algorithm to select relevant features and detect lane markers. Similarly, Kim and Lee [32] combine a Convolutional Neural Network (CNN) with the RANSAC algorithm to detect lanes starting from edge images. Huval et al. [29] show how existing CNN models can be used for highway driving applications. The end-to-end CNN that performs lane detection and classification at the same time. Li et al.[31] propose the use of a multi-task deep CNN that focuses on finding geometric lane attributes, such as location and orientation, together with a Recurrent Neural Network (RNN) that detects the lanes. However, most of them prone to limited to the highway scenarios, or they do not cope with the classification issue for finding various kinds of lane types.

To tackle this problem, we propose how a multi-task network can jointly handle lane and road marking detection and classification under adverse weather, low illumination conditions and difficult urban scenarios. Also, this can be stretched further to integrate ego-lane index identification network.

Table 3.1. Technology migration in lane-level localization.

Environment / Related work	Lane detection scope	Test bed	High way	Urban		Total lane number
				Narrow	Broad	
X. Du [23]	Find only driving lanes, Map matching	✓				1
R. Peixo [25]	Find only driving lanes, Map matching		✓			2
E. Jang [26]	Find only driving lanes, Map matching		✓			3
C. Guo [27]	Find only driving lanes, Graph-SLAM			✓		2
J. Shur [28]	Multi lanes, Unaware lane type, Map matching			✓		3
D. Cui [24]	Multi lanes, Aware lane type, Map matching			✓		3
Proposed method	Multi lanes, Aware lane type & ego-lane index, Map matching				✓	2-5

3.2.2 Lane-index Identification

Ego-lane index, defined in this paper, indicates which lane the ego-vehicle is driving. If the car is driving in third lane out of four-lane-road, it is good to classify the lane index, third lane, for lane-level localization or navigational system. There should be additional back-up system since there are one or two lane deviation from the real position occurs even in RTK-GPS. Researches using a camera sensor have been introduced because these are relatively inexpensive to solve this problem.

Kuhnl et al [7] proposes to utilize the spatial-ray features from a road's bird-eye-view (BEV) to infer the ego-vehicle's lane. The study contribution is that a novel feature can be adopted in road detection but its computation cost is somewhat heavy only for ego-lane recognition. *Seo et al.* [8] uses Bayes filter for multi-lane detection, and J. Rabe [38] utilizes particle ltering approaches. They proposed a tracking model of lateral localization using vehicle dynamics. On the other hand, A. Ballardini [36] derives ego-lane by HMM of lane tracking. Even if the authors in [7], [8], [36], [38] rely on solely using a camera, both studies focus only on highway scenarios.

Meanwhile, H. Cheng [37] identifies ego-lane index using image features and Baysian classifier while [20] also utilizes Baysian classifier with multiple evidence from the visual processing pipeline. C. Chen [35] proposed deep learning framework for finding lane index. This is end-to-end method without specifying lane features from

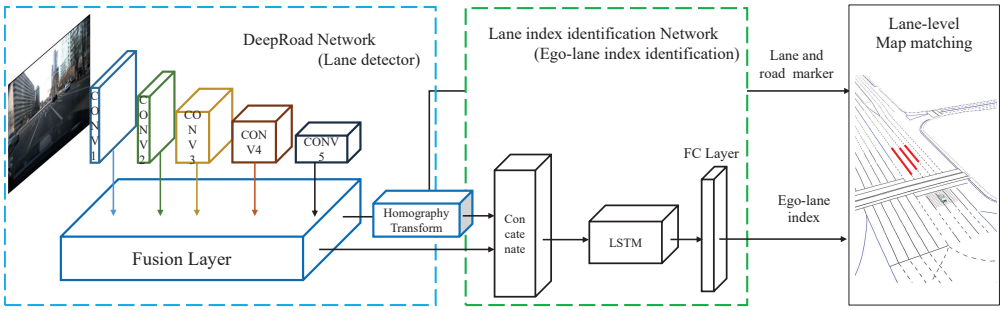


Figure 3.1. Ego-lane estimation network

the image. This method directly detects the ego-lane index without extracting neither lane markers, types, nor coefficient result and learns based on simple road conditions. Therefore, it shows the performance with narrow and limited road environments. These approaches prone to have problems with intersection, or broad roads wider than four-lane.

3.2.3 Lane marking based localization system

Road marking and lane based map matching has been introduced in various ways. Those methods are compared in table 3.1.

X. Du [23] has been proposed lane level map matching using only driving lane detection results. Therefore, their results is only tested in narrow road of their test bed site. It highly focusing the driving lane since the test environment have one lane. E. Jang [26] and R. Peixo [25] are handling map matching only driving lane detection and developed limited to the highway condition. On the other hand, mapping matches in urban scenarios using lane detection results are more difficult. A variety of environments such as lane shapes and characteristics, broken lanes, and intersections make this difficult.

C. Guo [27] has shown graph-slam based localization using lane detection results in urban scenario. However, their evaluation is performed in a small downtown area and is not considered any intersection area. J. Shur [28] has been propose multi-lane

map matching localization since most of them utilize only driving lane, i.e. single lane, for map matching based localization and it has less reliability when the result of driving lane is failed. The performance of their algorithm is considerably stable even in evaluation of heavily crowded urban scenarios.

However, this method only extract lane coefficient without lane type information and this would be less stable on large urban roads and they have not proven lane changing situation accordingly. On the other hand, D. Cui [24] proposed map matching method by adding lane type information as a complement to this weakness. Nevertheless, it appears that the algorithm does not consider the ego-lane index for localization and does not consider the intermittent break-lane measurement caused by intersection or crosswalks in urban conditions. To complement the limitations caused by these missing modules, we propose a new localization model that recognizes and integrates such missing information.

3.3 Framework

The proposed framework consists of integrated deepRoad network and map matching schema using the results of deepRoad network. Integrated DeepRoad net is comprised of DeepRoad net which detects road and lane marker detection and lane index identification which classifies ego-lane index. By using those recognized information in the map matching session, the ego-vehicle pose is estimated.

3.4 DeepRoad-Net: Lane Marker Detection

In this section, we propose a unified road and lane detection method within a single procedure. We extract lane and road markers using DeepRoad-Net and associate those markers to elicit lane and road information.

3.4.1 Network structure

The convolution neural network (CNN) labels pixel-perfect units by marking the label each image pixel contains. The basic architecture is similar to Holistically-Nested Edge Detection (HED-Net) [18], which is proposed for edge detection by modifying the VGGNet [19] architecture. The HED-Net attached the side output layer to the end of each convolution layer and deleted the last fully connected layer from the VGGNet.

Holistically-nested(HN) architecture adapts multi-scale which effectively consider both low-level, high-level feature. Most of the semantic segmentation network applies multi-scale concept inside the hidden layer(feature extraction level) which inspired by skip connection. Result of Superpixel Sampling Network shows this concept can learn multiple, complex shape of label. One distinctive characteristic of HN architecture from the other state-of-the-art semantic segmentation networks is applying multi-scale concept only onto the side output layer which contains actual label result. This turned out to be highly effective on labels which contains only simple features, in our case, various types of lanes, road markers. It also showed clear contour shape compared to the feature extraction level concatenation concept which showed noise so that it required other refining process such as Conditional Random Field.

However, whereas the HED-Net merely solves the binary classification problem corresponding to edge detection, our problem extends to the multi-categorization problem classifying lanes and multiple road marking types. Fig. 3.2 shows the network structure used for marker detection. After configuring the 1x1 side output layers with multi-road marker labels, the last output is derived through the 1x1 convolution layer.

3.4.2 Data Augmentation

We utilized BDD100K dataset [16] for data training. The dataset provides an order of magnitude larger than other state of the art dataset, and is comprised of over 100K videos with diverse kinds of annotations including image level tagging, such as drivable areas and lane markings. Among the training set, the labels are re-annotated in

Table 3.2. DeepRoad-Net architecture

layer name	output size	filter shape	side-output
conv1_3	480x320x7	5, stride 1	side out1
conv2_3	240x160x7	14, stride 2	side out2
conv3_3	120x80x7	40, stride 4	side out3
conv4_3	60x40x7	92, stride 8	side out4
conv5_3	30x20x7	196, stride 16	side out5
concate.	480x320x7		
Fc7	480x320x1		global out

7 main lane and road marker categories (road curb, double lane, single yellow, single white lane solid, single white lane dashed, stop line and road markers).

The input format of image can be different according to the viewpoint, based on the yaw, pitch and relative lane-vehicle position. Thus, we perform data augmentation by translating, pitch, and yaw changes from the acquired raw data, which increases the number of data samples by 9 times. Considering that captured road and lane images rarely show changes along a roll or height(z) direction in a typical driving environment, data augmentation involved in those aspects is not considered.

3.4.3 Implementation Detail

The loss function (L) applies the Softmax cross entropy function for all pixels. Each side output layer and final function output layer are defined in a combinational way. The total loss function is the sum of the concatenated loss L'_{fused} in the fusion layer and the loss of each convolutional layer $L^1_{side}, L^2_{side}, .., L^5_{side}$. Also, y refers to the Softmax output results of the network, t refers to the Ground Truth, n refers to the pixels and k refers to the marking label.

$$L = -\sum_{n}^{N_{Pixel}} \sum_{k}^{N_{label}} t_{nk} \ln(y_{nk}), L_{total} = \lambda \sum_{l}^{N_{side}} L^l_{side} + L^6_{fused} \tag{3.1}$$

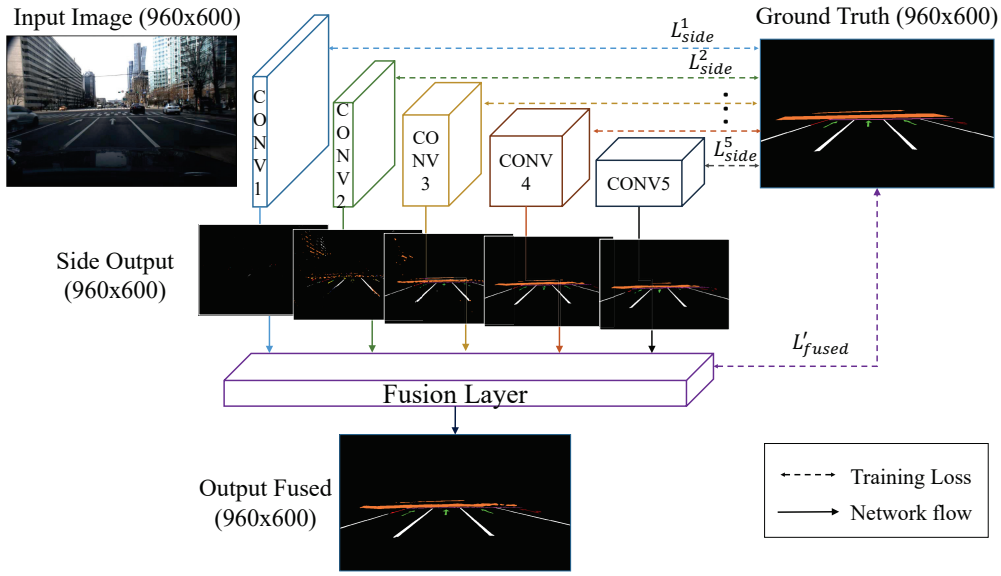


Figure 3.2. DeepRoad-Net

The size of the output of the network is the same as that of the input. It is similar to the well-known Fully Connected Network (FCN) in that a pixel-level output is presented.

For the use of training data, we have seven labels consist of various lane types, stop lines, and road information. We use grid-search on the validation set to select the hyper-parameters. We pre-trained our network on ImageNet [40], and employed a learning rate of 10^{-3} for fc7, and 10^{-6} for then rest when fine-tuning. Converging of the training requires approximately 200 epochs. Adam optimizer [42] is exploited with a L-2 distance based loss function.

3.4.4 Adaptive Birds Eye View Transform

Since the lane markers in the image is not efficient information, the homography matrix which transfer from image to the real local position is important. Because the detected lane markers in images are not efficient information for localization, a homography matrix for converting from image to actual local position is important. In our previous work [3], Dynamic Homography Matrix (DHM) for transforming Bird's Eye View

(BEV) as well as Cross ratio for robust lane position estimation is proposed. DHM H is derived by four source points from the driving lanes and four designated destination points with a consideration of real coordinates. H is updated consistently with every frame by calculating the equation as follows:

$$X = Hx,$$
$$\begin{bmatrix} X_{dst}W \\ Y_{dst}W \\ W \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x_{src} \\ y_{src} \\ 1 \end{bmatrix}, \tag{3.2}$$

where the source point is (x_{src}, y_{src}) and the destination point is (X_{dst}, X_{dst}). The cross-ratio is also defined as follows:

$$C(x_1, x_2, x_3, x_4) = \frac{|\overline{x_1x_2}|\,|\overline{x_3x_4}|}{|\overline{x_2x_4}|\,|\overline{x_1x_3}|}, \tag{3.3}$$

given four points x_i of lane detection results. The determinant of $\overline{x_ix_j}$,$|\overline{x_ix_j}|$, indicates real coordinate from x_i to x_j. Those values are gathered from measured distance from LIDAR sensor or predefined calibration value. The value of the cross-ratio is invariant, $C(x_1, x_2, x_3, x_4) = C(x'_1, x'_2, x'_3, x'_4)$, under any projective transformation. We can acquire an accurate bird eye view and real value of lane marker position. If we want to make deep neural system use in batches, the detected result can be also transformed to the birds eye view perspective can be derived by H-net [30].

3.4.5 Lane Marker Feature Clustering and Fitting

The detected lane markers using DeepRoad-Net is crude information. In order to extract meaningful information to localize the vehicle, it should be associated which lane the particular lane marking feature belongs to. In order to achieve it, we leverage Directional-DBSCAN[21] clustering algorithm for associating extracted lane markers. This clusters in each lane to account for the direction of the cluster. The lane model, $y = C_0 + C_1x + C_2x^2$, is represented by the second order polynomial equation which gives lane level lateral offsets C_0, heading angle C_1, and curvature of the lane C_2 [20].

3.5 Ego-lane index Identification

Ego-lane inference network utilizes the result of first network. After the result of road markers are achieved, the result is transformed to the birds eye view perspective using H-net. After the network, we obtain two images including birds eye view of lane detected result and a frontal view of lane detected result. Those two image results are concatenated as one input with the size of 56x56x2. The concatenated lane detection image set performed as an input for ego-lane convolution network. Output of the ego-lane index identification network is derived which lane they are driving, ego-lane index, after processing a fully connected layer.

3.5.1 Primal Input Data for the Architecture

Birds eye view of lane detected results from the consecutive frames are required to produce primal input data of the proposed DNN architecture. The output of the first network is result of multi-lane detection image which is relative pose between distinctive lanes and the current vehicle. Input of each image consist of detected lanes with different lane type such as yellow lane, double lane, dashed lane, solid lane or road boundary.

A bird eye view image of detected lanes on the road with different lane patterns is transformed to an discrete representation. Grid map is an discrete representation of front view from the ego-vehicle. Each grid cell contains spatial information of 10cm $\times$ 40cm. In this environment, lanes are divided into discrete grid of 240x160 pixels in total by dividing the area surrounding the vehicle in to 40m longitudinally, 24m laterally. The grid map composed of 7 labels, collected from the results of DeepRoad-net and birds eye view transform. The label index is defined that the ground presents label 0, while a dashed lane has label index 1, a solid lane with label index 2, yellow, double lane have other label 3, or road boundary presents label 4. Less important features containing longitudinal information is assigned: stop line label 5 and road marking label

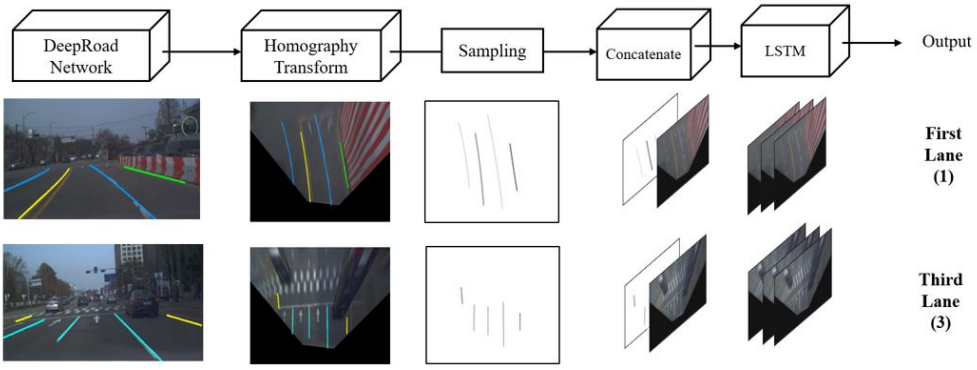


Figure 3.3. Input and outputs of each step in the Ego-lane Identification Network

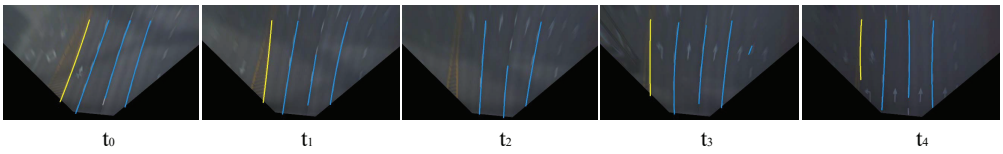


Figure 3.4. Input of ego-lane sequence images: case of driving in the second lane

6.

In order to compose the input for the LSTM network, Fig. 3.4 shows the sequential birds eye view of detected lanes with their types that are associated over k frames, where k is five in this study. The leftest image indicates a birds eye view lanes in the previous first frame at time t_0 and the time sequence is arranged from left to right in the figure. The rightest image presents current frame t_4. When it comes to the input sequence of t_2, the detection of yellow lane can be omitted because of true negative of the previous network.

3.5.2 Ego-lane Index Identification: Network Architecture

Second network in Fig. 3.1 presents for ego-lane identification network. Lane detected results containing lane characters in BEV perspective classifies ego-lane index. This can be derived in two methods, single network and RNN based sequential processing.

ConvNet for single image understanding

Unlike, sequential image concatenation is not performed in this method. Only singe current lane detection result is utilized. The network consists of five convolutional layers followed by three fully connected layers and a soft-max layers. The output of each individual layer can be extracted from the network and used as a holistic image descriptor. The three fully connected layer and the softmax layer are retrained to fit in to the lane information features specifically, since it is pretrained to the ILSVRC task (it have 1000 output neurons as like ILSVRC).

LSTM-based Sequential Understanding

Aforementioned network works at a single image frame. However, considering that the ego-lane index never changes dramatically, it depends highly on sequential information and it is appropriate to deploy Recurrent Neural Network(RNN) to derive much reliable and stable result. Because of steady sequential characteristic of ego-lane, this module does not require real-time computation over every frame. It is still okay to grab image sequence in every two-second. After acquiring 5 consecutive frames, those lane image inputs are concatenated and processed in the RNN network.

The lane detected formation observed over consecutive data frames can be trained by RNN layers. After receiving an activation sequence from the second CNN layer, e.g., $x = \{x_t | t = 0, ..., t = k - 1\}$, hidden state vector h^t , or activations from a conventional RNN layer, is computed as follows: $h^t = f(Wx^t + Uh^{t-1})$, where W and U are parameter matrices that need to be trained, and f is an activation function that is typically a logistic sigmoid function or a hyperbolic tangential function. Initial h^t is typically set as zeros. In this study, a LSTM layer is chosen because it showed slightly better performance in sequential accumulation compared to GNU layer. Core of the LSTM node is the memory cell which stores its state. Information stored in the memory cell is updated with the help of two gates. The 'Input gate' regulates the amount of information that gets added into the cell state, whereas the 'Forget gate'

Table 3.3. Index identification network architecture

Layer name	Output size
Lane detection layer (lane type label)	480x320x7
Top-view Transform	480x320x1
Lane type sampling (Resize)	240x160x1
Concatenation (Sampled & resized original gray image)	240x160x2
LSTM	50x1
FC1	128x1
FC2	5x1

controls the amount of information that is removed from the current cell state. The input, forget and output gates use the sigmoid activation. The hyperbolic tangent of the cell state, controlled by the 'output gate' is given as the output of the node.

$$
\begin{aligned}
a^t &= \tanh(W_c x^t + U_c h^{t-1} + b_c) \\
i^t &= \sigma(W_i x^t + U_i h^{t-1} + b_i) \\
f^t &= \sigma(W_o x^t + U_o h^{t-1} + b_o) \\
c^t &= i^t a^t + f^t c^{t-1} \\
h^t &= o^t \tanh(c^t)
\end{aligned}
\tag{3.4}
$$

The equations above describe how a LSTM cell is updated at every time-step t. In these equations, x^t is the input to the memory cell at time t. i^t, f^t and o^t are the input, forget and output gate values at time t. a^t is the modulated input to the cell at time t. c^t denotes the cell state value at time t, where as c^{t-1} is the cell state value in the previous time step. h^t is the output of the cell at time t. Because of this particular node structure, LSTMs are able to store the information for longer time-steps and hence can solve the vanishing gradient problem. This makes LSTMs a popular and powerful choice of ego-lane identification, where there is a need to store long contextual information.

3.5.3 Data Augmentation

Since birds eye view image is the input format of our deep neural network, it can be augmented according to the slight viewpoint variance. Thus, we perform data augmentation by rotating the acquired top view images by 15 degrees each along a yaw direction, and by translating data under x, y coordinate that lane keeps the same lane index. Since the pitch changes in acceleration and deceleration of the vehicle can generate a few distorted BEV images, the BEV images with slight pitch errors are additionally augmented. This increases the number of data samples by 10 times. Considering that detected lane rarely show behavior along a roll direction in a typical driving environment, data augmentation involved in those directions is not performed.

3.5.4 Implementation Detail

For classification, the output labels are annotated in 5 lane index categories (center lane, 2nd lane, 3rd lane, 4th lane, outer lane). As a loss function, we utilized categorical cross entropy in the following form:

The loss function (L_i) applies the Soft-max cross entropy function for all lane index. Fully connected layer provides classified ouput y_i. The loss function is the Soft-max classifier that uses the cross-entropy loss:

$$L(p, q) = -\sum p(x) log(q(x)),$$

where p(x) is the probability of a ground truth label that is typically a one-hot vector and q(x) is the probability of a predicted label computed by the soft-max classifier of the last FC layer. Fifty percent of node drop out is adapted and Adam optimizer was employed to optimize the loss function. The initial learning rate was set to 0.005 and the number of epochs was set to 1000.

3.6 Localization using Extracted Lane and Lane-index Information

When we receive data from cameras, lane, road markers and identified lane-index are detected using deep neural network and post processing in the aforementioned perception sections. After obtaining both lane coefficients and lane index, there are two types of vehicle positioning. First method, local lane keeping-level positioning, is to localize within the lane-level. This works at the highway conditions under the limited map information such like a navigational map. The centimeter-level longitudinal accuracy is less required since lane keeping is mostly enough until the target ramp appears. On the other hand, global positioning using lane-level map matching is more a good alternative since the road environment is more complicated and this requires accurate localization in urban scenarios.

3.6.1 Local Positioning

After lane information including lane coefficient and lane index is decided, the system can locally position ego-vehicle. We suppose the shape of the lane is maintained and the vehicle is travelling along the lane. This will allow the relative motion of the vehicle and its lane to be modeled. The vehicle has a continuous motion with speed and yaw-motor information, so the relative indicators of vehicle signals and lanes can be expressed as shown in equation below,

$$\begin{aligned} C_0 &= e_y = y - y_{des} \\ C_1 &= e_{yaw} = \psi_{yaw} - \psi_{des} \end{aligned} \tag{3.5}$$

We also assume that the time variation in lane coefficients is linear. Since the coefficient C value in time $t+1$ can be expressed as the product of time t and coefficient $\frac{dC(t)}{dt}$ in the value of the coefficient in time t, it is possible to derive the relationship

between time and the for each lane coefficient based on equation,

$$C_{t+1} = C_t + T_c \dot{C}_t. \tag{3.6}$$

When aforementioned two equations are used to derive the lane prediction model, and express it in a state space model, the equation is derived as shown in follow:

$$\mathbf{x} = \begin{bmatrix} C_0 & \dot{C}_0 & C_1 & C_2 \end{bmatrix}^T, \mathbf{u} = \begin{bmatrix} a_y & \dot{\psi} \end{bmatrix}^T \tag{3.7}$$

$$\begin{aligned} &\mathbf{x}[k+1] = A\mathbf{x}[k] + B\mathbf{u}[k] \\ &A = \begin{bmatrix} 1 & T_c & 0 & 0 \\ 0 & 1 & -a_x T_c & -2V_x^2 T_c \\ 0 & 0 & 1 & 2V_x T_c \\ 0 & 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ T_c & V_x T_c \\ 0 & -T_c \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{3.8}$$

$$\begin{aligned} &\mathbf{z}[k] = H\mathbf{x}[k] + \mathbf{v}[k] \\ &H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.9}$$

$$\begin{aligned} &\mathbf{x}[k]^- = A\mathbf{x}[k-1] + B\mathbf{u}[k-1] \\ &P_k^- = AP_{k-1}A^T + Q \\ &K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \\ &\mathbf{x}[k] = \mathbf{x}[k]^- + K_k(z[k] - H\mathbf{x}[k]^-) \\ &P_k = P_k^- - K_k H P_k^- \end{aligned} \tag{3.10}$$

Based on the Kalman filter, the stabilized lane coefficients are derived. Using those C_0 and C_1, lateral offset, i.e., lateral localization can be obtained concerning ego-lane index L_{index}.

$$\begin{aligned} &L_{Dist_from_center} = w(L_{index} - 1) + C_0 \\ &L_{orientation} = \arctan(C_1) \end{aligned} \tag{3.11}$$

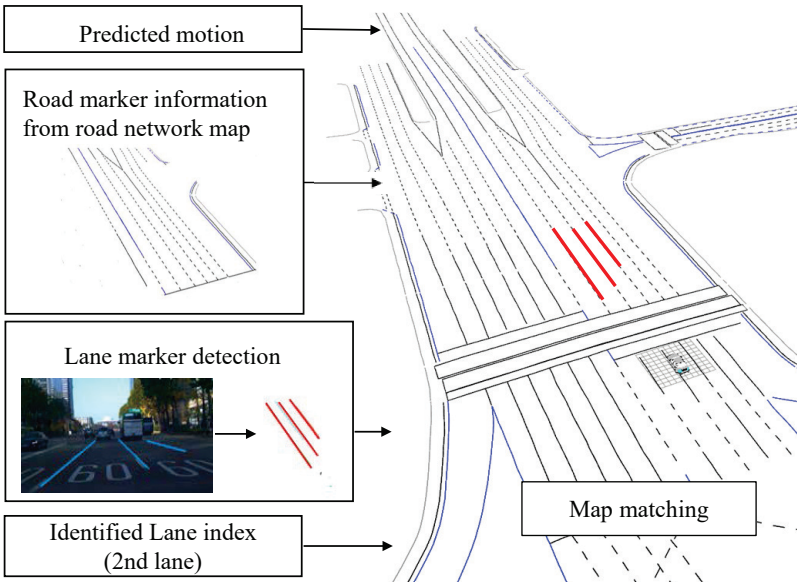


Figure 3.5. Map-matching lane

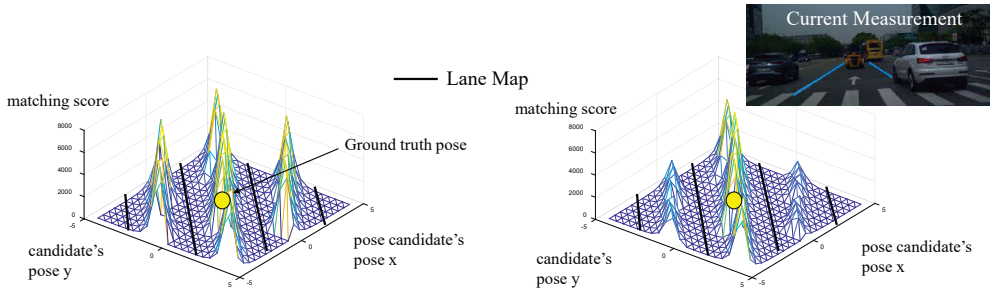
3.6.2 Global Positioning: Road Network Map Matching

When we receive data from cameras, lane, road markers and identified ego-lane index are detected using deep neural network and post processing in the aforementioned perception sections. We compare the detected lanes and road markers with road marker information from the road network map to estimate the precise position of the vehicle. In global positioning aspect, we adapt the road network map for map matching and the vehicle is globally positioned in the road network map.

Motion Update

The state of next time step is estimated based on the system model. A vehicle kinematic model with on-board sensor inputs is used for time updates as shown in eq. 3.12.

$$\hat{\mathbf{x}}^i_{k+1} = \begin{bmatrix} \hat{x}^i_{k+1} \\ \hat{y}^i_{k+1} \\ \hat{\varphi}^i_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{x}^i_k + \tilde{\nu}\cos(\hat{\varphi}^i_k + \tilde{\omega}\Delta t) \\ \hat{y}^i_k + \tilde{\nu}\sin(\hat{\varphi}^i_k + \tilde{\omega}\Delta t) \\ \hat{\varphi}^i_k + \tilde{\omega}\Delta t \end{bmatrix} \tag{3.12}$$



[Lane index un-aware lane map matching result]

[Lane-index AWARE lane map matching result]

Figure 3.6. Measurement scores($w^N_{Lane,k}$) of given pose candidates ($\mathbf{x}^N_k$) whether ego-lane index information is given or not. Pose (0,0) presents the actual pose. Black line presents local lane maps.

The control input of the vehicle model can be obtained from the vehicle motion sensors: a yaw rate sensor w_{gyro} and wheel speed sensors V_{whl}. We can predict the next step's position candidates where ϵ_v and ϵ_g are gaussian random variables. The shape of the probabilistic distribution can be changed via the configuration of sensor noises. $\tilde{\nu} = V_{whl} + \epsilon_v, \quad \tilde{\omega} = \omega_{gyro} + \epsilon_g$

Map Information from Road Network

We assume an environment in which road networks are pre-built. Our road network map is labelled based on ID. Map information is easy to read directly based on the structured network data. According to the road network structure, connection relationship among each other features is structured by id of link, which is the center of the lanes. For example, each lane has attribute information which is the id of the link on the left and the id of the link on the right. Id also has information on the index number of lanes and information about the color and shape of the lane. Similarly, the road mark also has the corresponding link's id. Therefore, corresponding link is loaded and the attributed information such as drivable path, current lane, center lane, lane index, and road markers from the link are loaded when the current pose is given.

Measurement Update

For measurement update, each weight is updated by evaluating the likelihood of map matching measurements $p(z_k|x_k^i, m_k)$ conditioned on the estimated i-th particle state x_n, sensing measurement z_k and the local map m_k at time stamp k. The relationship between the weight and likelihood can be expressed as $w_k^i \propto w_{k-1}^i \cdot p(z_k|x_k^i, m_k)$. where, $k-1$ is the matching weight at previous time and $p(z_k^i|x_k^i, m_k)$ is the current measurement.

To simplify the calculation, the final weight of the particle w_k is determined as $w_k^i = w_{k-1}^i \cdot w_{Index,k}^i \cdot w_{Lane,k}^i$, where $w_{Lane,k}$ is shown as follows:

$$w_{Lane,k}^i = w_{Lane,k-1}^i \cdot e^{-\frac{(d_k^i - C_{0,Lane})^2}{2\sigma_d^2}} \cdot e^{-\frac{(\theta_k^i - C_{1,Lane})^2}{2\sigma_\theta^2}}. \tag{3.13}$$

where linear line coefficient $C_{0,Lane}$ and $C_{1,Lane}$ are current lane measurement and θ_k^i and d_k^i are relative orientation and distance between the given local map m_k and sensor z_k conditioned on i-th particle pose state. On the other hand, ego-lane index measurement update model $w_{Index,k}^i$ is shown as follows:

$$w_{Index,k}^i = w_{Index,k-1}^i \cdot e^{-\alpha|L_{index} - l_k^i|}, \tag{3.14}$$

where L_{index} is the output result of ego-lane index identification network and l_k^i presents lane index of i-th particle pose. Every output result has lane index L_{index} and its belief probability from Soft-max classifier. If the ego-lane output result has low belief probability lesser than substantial threshold, this index model $w_{Index,k}^i$ is not used and set to 1 because of belief insufficiency.

Each map matching weight is calculated at the heterogeneous matching step. As for all calculated final weights $w_k^i (i \in 1, ..., N)$, w_k^i is normalized by equation $\hat{w}_k^i = w_k^i / \sum_{\mathrm{j}=1}^{\mathrm{N}} w_k^j$, where N is the number of particles. This normalization would resample the sum of final measurement probabilities to one.

State Estimation and Re-sampling

The final position of ego-vehicle is to be determined by maximum a-posteriori (MAP) estimation, which the estimated pose $\hat{x}_k$ can be approximated as follows:

$$\hat{x}_k = \underset{\hat{x}_k^i}{argmax}\, p(x_k^i | z_k, m_k), \tag{3.15}$$

where local map m_k and road semantic measurement z_k are given. Resampling re-generates a new set of uniformly generated candidates on the basis of their relative likelihoods. If no fault is detected during the probabilistic situation analysis phase, the resampling phase diffuses several concentrated candidates at the next step. If a dissimilarity or a fault is detected, the next step increases the size and interval of the candidates to search for a larger area with uncertainty.

3.7 Experiment and Evaluation

The contributions of this study include the evaluation of three factors: the lane detection performances, ego lane index identification performance, and ego-lane index aware localization performances.

3.7.1 Experimental Set-up

We tested two scenarios urban and highway to show the robustness of our proposed algorithm. Video clips were taken from diverse places in large city, Seoul, Korea as well as inter-city route and highway to satellite cities. An Edmund camera, *EO-0813C*, with a 6mm lens and a downsized resolution 480×320, was used. Our proposed algorithm is implemented using OpenCV, TensorFlow and TensorPack libraries for lane detection and lane index identification model. The localization algorithm is implemented and tested using C++/OpenCV/TensorFlow and operated using ROS. The algorithm runs at a frame rate of 20.1 fps on a normal PC (Intel Core i5 Q720 CPU @ 2.2 GHz, 12GB RAM and GeForce GTX 1080 Ti). Each step's computing cost results

in DeepRoad-Net for 13ms including marking detection, ego-lane index identification for 10ms, post-processing for 12 ms and road network map-matching for 25 ms. Since lane marker extraction and map matching processes are processed in a sequential manner, and ego-lane identification and lane post processing are performed in parallel, the total processing time is 50 ms. It meets considerably real-time requirement assuming that 100 ms for autonomous driving as the minimum requirement, corresponding to the overall vehicular environment sensor measurement update cycle.

3.7.2 Evaluation on Lane Detection Performance

Evaluation Metric

Lane marker extraction performances is compared with other popular algorithms. We evaluate 500 images containing different illuminance conditions, including shiny, rainy, night conditions. We utilise a graphic software tool and label the line markings manually to build ground truth. Three classic metrics [45] are calculated as:

$$Recall = \frac{\text{True positive(TP)}}{\text{True positive (TP)} + \text{False negative (FN)}}, \tag{3.16}$$

$$Precision = \frac{\text{True positive (TP)}}{\text{True positive (TP)} + \text{False positive (FP)}}. \tag{3.17}$$

$$MaxF = 2 \times \frac{precision \times recall}{precision + recall} \tag{3.18}$$

Evaluation of Various Back-bone Network Utilization

To find optimal performance, either VGGnet, ICNet [43], or MobileNets [44] can be applied as a back-bone network for convolution steps. We compared these networks' performance as shown in Table 3.4. To compare each network in depth, evaluation in this table is to solely evaluate road marker extraction results, except for the results of post-process lane. This evaluation is done by pixel level assessment.

In the Table, the better results can be achieved by more sophisticated pixels with larger image with the same VGG-Net. However, the processing time is obviously not

Table 3.4. Performance Comparison in Various Network Structures.

Network	VGG-Net		ICNet	MobileNets
	(480x320)	(960x640)	(480x320)	(480x320)
Recall	0.842	**0.874**	0.865	0.867
Precision	0.896	0.910	0.892	**0.912**
Proc. Time(ms)	35.1	105.4	27.8	**12.7**

feasible to the real-time application, we found the 480x320 size optimal. As well as the size, this tabel shows that original HED network using VGG-Net is not ideal when comparing other back-bone network's performance. MobileNet have faster processing times while sustaining substantial performance. As aforementioned, MobileNet is cost-effective with the fairly good performance. From now on, other evaluations using the proposed lane detection algorithms utilize the MobileNet as the backbone network.

Performance Comparison to Other Methods

This part is to evaluate overall lane detection algorithm performance. This includes post-processing part of lane association exploiting lane coefficients. As shown in Fig. 3.7, the result of our lane detection is shown below. This figure presents lane detection results with distinguished lanes such as yellow lane, solid lane, dashed lane and road boundary. Each color describes the lane meanings as shown in the center of the figure. We compare the performances using KITTI dataset images [41] but did not follow the metric of them since their method is confined to the driving lane regions. Therefore, we derive the performance with following *Kim*'s method [30]. The ground truth target includes all lane instance within 40m FOV. Within the FOV, if the detected lane is positioned 4-pixel away from the ground truth, this detected lane is counted as false positive. Metric of recall and precision follows the definition above.

Table 3.5. Performance Comparison to Other Detectors: KITTI Dataset

	Image processing			Deep learning	
	CRF[12]	SPRAY[7]	Const. VP[34]	LaneNet[30]	Ours
Recall	0.897	0.886	0.931	**0.941**	0.921
Precision	0.883	0.877	0.918	0.936	**0.976**
MaxF	0.890	0.881	0.924	0.931	**0.948**

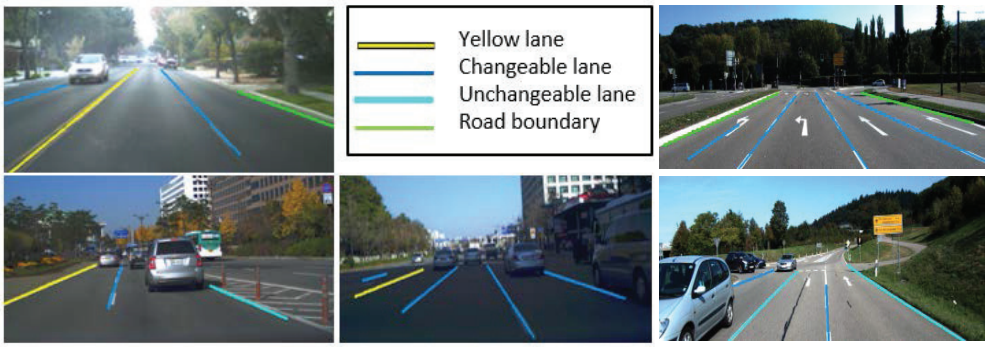


Figure 3.7. Lane detection results

Obviously the deep neural network based algorithm have better performance in most of the cases. Compared to LaneNet, our proposed lane detection algorithm have lower results in recall while having higher precision. This pattern has pros and cons. Considering both ego-lane index identification and lane-based map matching have better performance in lower false positive, our model is considerably efficient in relevant algorithm extension using the detection results.

3.7.3 Evaluations on Ego Lane-index Identification

Dataset

Since this research topic does not have public dataset, all data has been tested within our data. We utilise a stabilized point cloud map localization system and road network

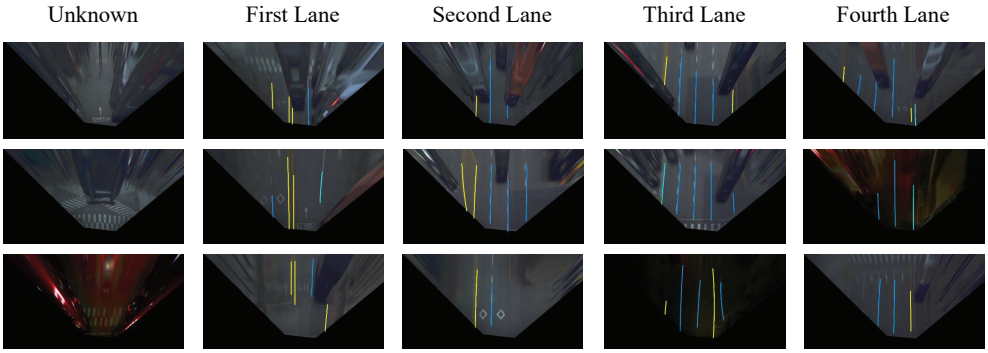


Figure 3.8. Ego lane results

map to collect ground truth. Within the proper vehicle position, road network map provide the ego-lane index value. The derived ego-lane index and corresponding image sequences are gathered as ground truth, categorizing them as training set and test set.

We evaluate 6600 images containing harsh weather conditions such as shiny, rainy, night time conditions. A total tested image can be also categorized by 2,000 frames of highway scene and 4,600 frames of urban scenes. In both conditions, tests of 2, 3, 4 and 5 lane-roadway are evenly performed.

Evaluation

Various types of lane driving circumstances are evaluated such like in classified results in Figure 3.8 to show our ego-lane index identification algorithm performance. In this chapter, classification accuracy is evaluated as lane index identification is one of the classification problem.

$$Accuracy = \frac{the\ number\ of\ correctly\ identified\ lane\ index\ cases}{the\ number\ of\ all\ tested\ cases}. \quad (3.19)$$

The classification accuracy is the percentage of correctly identified lane-index images out of the number of all tested images, is an evaluation metric in this part. Our algorithm is evaluated and compared to other previous algorithms in Table 3.6. This comparison is performed within urban image dataset from camera logging data of

Yeouido A and B. In the table, broad roads are defined as four lanes or more, while roads having three lanes or lesser are defined as narrow road. The proposed algorithm has merits in urban region since it has more complicated road-lane cases such as merging, splitting or broken lane. End-to-end based lane index identification has lower performances in broad region. This end-to-end concept does not properly extract the feature of the center lane when the training data contains both one-way and bi-directional way image at the same time. Therefore, it works proper in narrow region but it does not react the broad region. Moreover, since Bayesian network [20] algorithm does not utilized lateral offset from the lane, it has weaknesses of lane changing situation in most of the cases. In this same problem, the proposed CNN based algorithm has more reliable recognition but still have less amount of stability in middle lane identification out of broad road. Meanwhile, the proposed LSTM based algorithm shows stable results in broad lane roads.

Table 3.6. Ego-lane index classification accuracy of various identifiers

Model / Case	Bayes Classifier [20]	End-to-end method [35]	Proposed (CNN-based)	Proposed (LSTM-based)
Highway	0.952	0.971	**0.981**	0.979
Urban Roads	0.894	0.931	0.945	**0.963**

Moreover, we compare the performance stability on diverse weather conditions. As shown in Figure. 3.9, we tested shiny, cloudy, night, and rainy conditions. Index identification results are relevant to lane detection results in Table 3.7. Therefore, this may results in consuming more time to have more sophisticated index classification results if the lane type recognition performance is reduced in the very harsh conditions. As long as the lane is well found, however, there is no significant effect on the ego-lane identification module. It was found that there was no degradation of performance on the data in which more than or equal to three lanes were detected.

On the other hand, in order to evaluate the sustainability on driving lane conditions,

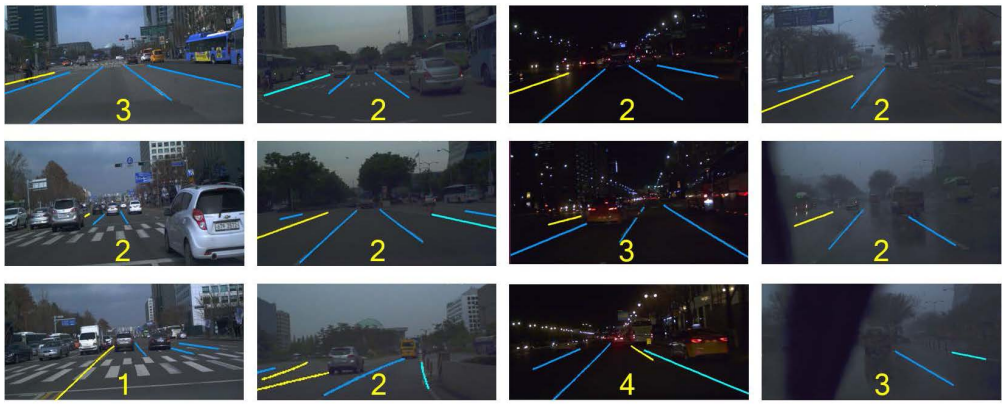


Figure 3.9. Lane and Ego-lane index results from various dataset scenarios: (a) Shiny, (b) Cloudy, (c) Night, (d) Rainy conditions

Table 3.7. Lane-index identification in various weather conditions

Function(Metric) \ Environment	Sunny	Cloudy	Night time	Rainy
Lane detection(MaxF)	0.966	0.963	0.941	0.929
Lane-index identification (Classi. Accuracy)	0.976	0.974	0.961	0.954

our LSTM-based ego-lane index identification algorithm is evaluated on each lane-index-level out of five lanes in total. Among the roads having different number of lanes, the results of driving on each lane position from the first lane to the outermost lane are shown in Table 3.8.

Obvious factors of this evaluation is that middle lanes such as second or third lanes out of wide road have less confidence since there are less distinctive lanes such as yellow lane, double lane, or road boundary. The proposed ego-lane recognition algorithm has an outstanding performance in broad road. the result showed less confident with five-lane roadway scenarios but we still achieved remarkable classification performance considering that other researches have not been considered before.

Table 3.8. Performances of each lane driving conditions (Accuracy).

Scenarios / Total lane	Driving Ego-vehicle's lane index					Average
	1st	2nd	3rd	4th	5th	Performance
2-lane-road	0.981	0.988				0.986
3-lane-road	0.971	0.976	0.969			0.973
4-lane-road	0.970	0.959	0.962	0.970		0.967
5-lane-road	0.965	0.960	0.940	0.945	0.967	0.955

3.7.4 Evaluation on localization performance

Test Datatset

We conducted an experiment on an 5 km zone in a wide urban environment up to 5 lanes in Seoul, Korea to assess the system's performance. Figure 3.10 shows a digital map depicting a map of the Yeouido road network and shows the vehicle trajectory in red. We verify the effectiveness of introducing ego-lane index with experiments on the Yeouido dataset. Dataset A and B with different paths are gathered to show various road conditions respectively. To evaluate the accuracy of the proposed localization system, a ground truth is assumed as a dense 3D point cloud map matching based localization results with aid of kinematic GPS (RTK-GPS). This shows 5.6cm error module and is performed as ground truth.

Evaluation

Localization performance is evaluated. We calculated the position errors from ground truth, achieving root mean square (RMS) errors. The results indicate that the ego-lane index information compensates for improper lane index placement situation and reduces the total amount of lateral offset errors. A comparison study is performed between the proposed algorithm additionally using ego-lane index and localization system only utilizing road marking based map matching. Based on the test dataset,

Figure 3.10. Yeouido Datatset.

we determined the absolute position errors, including lateral and longitudinal error, by differencing ground truth and measured positions using the proposed algorithm as shown in Table 4.5.

Table 3.9. Localization Performances: Mean (Standard dev.)

Scenario	Dataset A			Dataset B		
Accuracy / Methods	Lateral(m)	Distance(m)	Orientation(°)	Lateral(m)	Distance(m)	Orientation(°)
GPS	9.09 (8.10)	18.12 (16.26)	0.52 (0.32)	0.79 (0.43)	2.38 (1.32)	0.76 (0.41)
Lane-level Map matching [28]	0.63 (0.82)	1.62 (1.86)	0.39 (0.29)	0.28 (0.32)	1.35 (1.31)	**0.45** (0.27)
Ego-lane aware (Proposed)	**0.21** (0.23)	**1.26** (1.24)	**0.36** (0.30)	**0.24** (0.25)	**1.33** (1.36)	0.47 (0.31)

To better illustrate the importance of ego-lane index, a specific part of the driving data in which the car crosses the intersection is further described in Figure 3.11. On

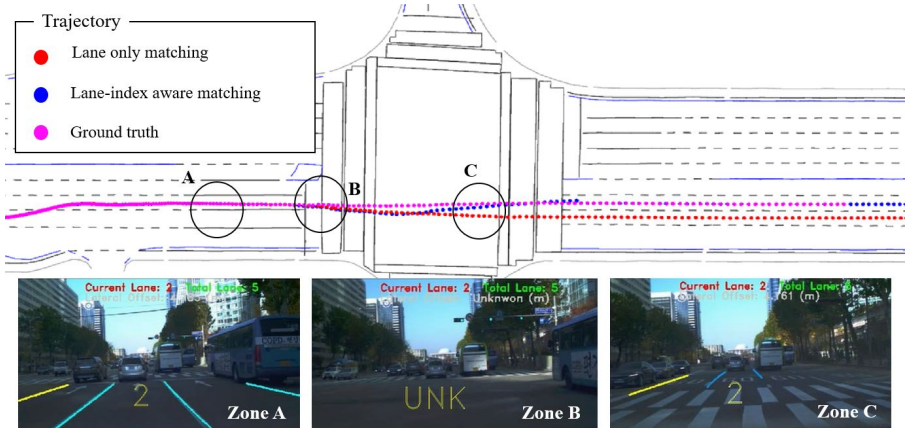


Figure 3.11. Ego-lane index-aware localization performance within intersection region

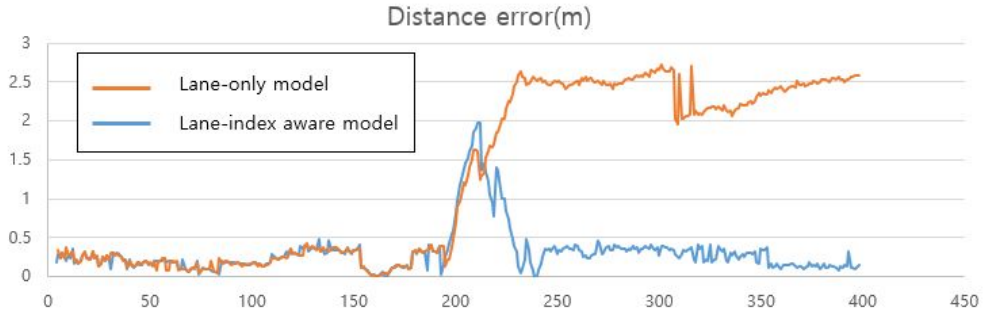


Figure 3.12. Errors of the localization results in Figure 3.11

the map in Figure 3.11, the test vehicle is maneuvering from zone A to zone C, which is from right to left. The output trajectory of each case is shown in the map. Red color trajectory indicates lane only matching method. Blue color indicates lane-index aware lane-based map matching, while magenta color presents ground truth trajectory.

Lane-based localization is normally less reliable to the global position information compared to in-lane localization. That is, the model is confused with adjacent lanes as it is similar to adjacent lane results where it is located in the middle of a wide roadway even though this localization within the lane is well performed. When entering zone B, there is no lane measurement in the middle of the intersection in the camera's view. In this section, the measurement is simply the vehicle dynamics of the IMU sensor without camera sensing and only positional estimation is carried out. Coincidentally, IMU information contains slightly contaminated data, resulting in a cumulative lateral error in this test case. The pose error mis-estimates that the vehicle is in the third lane during this session, but the actual pose is in the second lane.

At this moment, the lane map matching algorithm and the proposed algorithm work differently. Lane-only map matching algorithm focuses on matching between map and detected lane markings, and works well in positioning within the driving lane. Therefore, this matching method maintains the lane accuracy within the previous lane. However, this is only lane keeping pose estimation in which the first deviation has already occurred. Therefore, the distance error is not converged and have one-lane deviation as shown in Figure 3.12. Contrary to the pattern of distance error, the pattern of orientation error shows that the two algorithms are somewhat consistent. This means that even if the lateral offset includes errors, the orientation information within the lane is correct because it matches multiple lanes' orientation properly.

Meanwhile, the proposed algorithm informs the ego-lane index identification result that they are actually driving in the second lane. Therefore, this method gives higher matching scores to the particle in the middle of the second lane. Therefore, the matching result is gradually relocated to the second lane. Ultimately, as the lane moves

to the right lane position, it converged to right pose and contains little deviation to the ground truth.

Through this detailed analysis, ego-lane aware lane based matching is able to prevent a larger error from occurring by preventing the unit in the lane from failing rather than non-aware model. Not only this big error compensation, but the knowledge of the ego lane index information also confirmed that the overall performance is slightly more stable, in that it can bring map lane values more clearly on the road network map.

3.8 Conclusions

In this paper, we propose an ego-lane index-aware map matching based localization framework that utilizes road markers and ego-lane index to overcome improper lane index placement situation which happens a lot in lane-level map matching technique. Lane index is derived by integrated DeepRoad-net which provides road marker detection and ego-lane index identification at the same time. The next step will be a pose estimation network with advanced DeepRoad networks that directly solve the problem within a single network framework of lane-based localization.

Chapter 4

Directional-DBSCAN: Road-line Instance Clustering Method for Parking Slot and Lane Detection

4.1 Introduction

Lane detection or parking slot detection algorithms using visual sensors have been required for positioning ego-vehicle both locally and globally. The improvement of road marker detection system is significantly required since lane detection performance affects the lane based localization system in previous chapters. Among the procedures of lane and road marker detection, road marker feature extraction and feature association are main key for exploiting the results. Since the method of feature extraction is proposed in chapter 2, we focus on how to associate marking features within the same lane, parking line and road marking instance. In most popular feature association for detecting parking lines and multi-lanes, such as the Hough line detector or DBSCAN algorithm, have been employed. However, these algorithms are originally designed to find distinct features and are inadequate for the short, curvy, faint and distorted parking lines of long-range surround-view images. Not only that, thsoe feature clustering algorithm such as DBSCAN is ineffective in merging or split lines such as in line.

Among various kinds of vehicular environment perception application, we proceed

this chapter on focusing parking-line detection application majorly since it has more sophisticated patterns such as line merged and splitting shapes compared to the multi-lane detection application.

In this parking-line detection application, a large number of automated parking assistant systems (PASs) have been proposed to maximize both the safety and convenience of parking. The selection of a particular parking slot among available parking spaces is a primary component in a parking assistant system. Perception methods for available parking space detection have been developed mainly in two ways: The free-space-based approach [74]-[79] and the parking-slot-marking-based approach [64]-[73].

In the free-space-based approach, different types of sensors with ranging capability (e.g. a laser scanner and ultrasonic sensors) have typically been employed. This approach guarantees stable measurement, but is highly dependent on the existence and pose of adjacent vehicles. On the other hand, in the marking-based approach, slot markings are the main means of finding parking spaces and provide accurate positions and heading angles. With this information, parking becomes more accurate within a parking space. For these reasons, the slot-marking-based approach has been the focus of recent research rather than the free-space-based approach. Moreover, the marking-based approach has become more influential for PASs since the wider field-of-view (FOV) has become available with multiple visual sensor configurations, such as around-view monitoring (AVM) systems, in recent few years.

A variety of parking slot detection methods via the slot-marking-based approach have been proposed. With the aspects of feature utilization, the methods can be categorized into two types: Slot-corner-detection-based [68]-[73] and line-detection-based methods [64]-[66]. The corner-detection-based method using FAST corner[68] and Harris corner[64][65] is used to find the junction points of parking slots. *Suhr et al.* [69]-[71] utilize corner features to recognize slot patterns and parking spaces. By suggesting a hierarchical classifier considering corner angles, junction types, and parking

Table 4.1. Technology migration in parking-slot detection.

Module \ Method	Corner-based	Line-based	
Instance Feature Extraction	Harris/Fast	Detector	Clustering
	[68]-[71]	[64]-[66]	Ours
Pattern Recognition	[69]-[70]	Ours	

slot patterns, various kinds of parking slots can be potentially handled. Despite decent performances, these approaches have various limitations. First, corner features cannot always provide precise target positions and heading angles. In addition, since it is difficult for a corner feature to maintain robustness, these algorithms contain excessive constraints in scenarios when 1) multiple frames and slots are required to make a decision, 2) the entering side of slots should be visible, and 3) targets are confined to predefined patterns.

Meanwhile, the parking-line detection-based method provides an accurate direction and has fewer constraints compared with corner-detection-based methods. In addition, it is robust on occlusions caused by an ego-vehicle or objects. In the previous study, line detectors, including the Hough transform (HT) [64][65] or distance transform [73], were utilized to find parking lines. However, conventional line-detection-based methods mostly target a limited number of slots with a narrow FOV. Consequently, the direct adaptation of these methods in AVM images is fragile for short, curvy, and distorted lines, as shown in Fig.4.1-(a). Most importantly, these methods are confined to the simple rectangular slot type in the absence of slot-pattern recognition.

In this paper, we present a parking-slot detection algorithm using the parking-line clustering method. The proposed algorithm consists of parking-line segment detection based on directional density-based spatial clustering (Directional-DBSCAN/D-

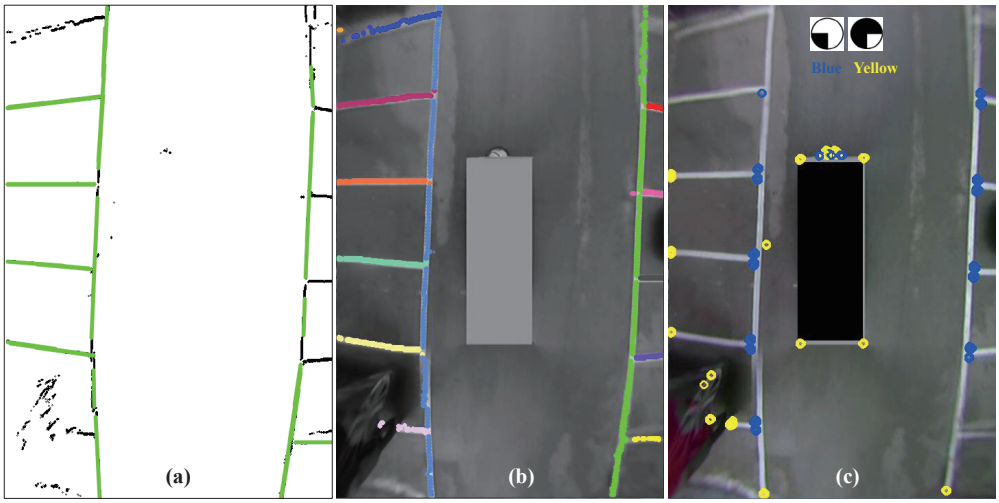


Figure 4.1. Difficulties in parking-line detection on distorted, short, and curvy lines: With same marking features (black), (a) Probabilistic HT detector (green)[65] missed many short lines, and long curved lines are snapped compared with (b) proposed D-DBSCAN results. (c) Corner detector[69] is also weak at blurry and distorted angles.

DBSCAN) and parking-slot detection with slot-pattern recognition by analyzing line segments. The main contributions of this study can be summarized as follows:

- We propose a D-DBSCAN clustering algorithm based on DBSCAN[63], but manipulated to be suitable for parking-slot detection applications. The proposed D-DBSCAN not only follows the original attributes (i.e. density reachability and density connection), but it also validates the orientation similarity of adjacent points. As a result, each orthogonal or slanted line segment can be clustered individually. The proposed D-DBSCAN-based line detection has higher capabilities in handling curved, distorted, and short parking-line marks compared with other line detectors.

- We also propose a slot-pattern recognition method based on a jointly combined Bayesian decision tree classifier analyzing line segments. By utilizing the geometric characteristics of the line segments, various types of parking slots are identified. The recognition of slot patterns has the following benefits. First,

parking-slot detection can be more robust by utilizing the individual characteristics of slot patterns and facilitating outlier elimination involved in light refection, non-parking landmarks, and objects' line edges. Additionally, the vehicle maneuvering strategy can be altered in accordance with the recognized slot pattern.

- We demonstrate the robustness of the proposed algorithm in diverse types of scenarios. For parking slot detection, irregular parking slot patterns, which have been barely tried, such as a rectangular parallel and open slanted patterns, are additionally included. For lane detection, we evaluate various merging and splitting lane Based on the harsh conditions including harsh weather scenarios and indoor scenarios, we demonstrate the robustness of our algorithms. Also, in order to show localization performance enhancement, we compared the proposed clustering instance based map matching algorithm to the raw map matching algorithm.

The remainder of this paper is organized as follows. Section II presents the parking-line segment detection stage using D-DBSCAN. The step is comprised of line-marking extraction, D-DBSCAN clustering, and line fitting. The slot-pattern recognition and detection stage, which consists of geometric feature extraction, pattern classification, and slot detection, is introduced in Section III. Section IV demonstrates experimental results and an evaluation of the proposed algorithm. Lastly, Section V draws conclusions.

4.2 Road Marking Association using Directional-DBSCAN

This section shows how parking-line-segments are extracted through the proposed clustering method. Accurate and robust parking-line detection plays an important role in two aspects. The recognition result is utilized not only to localize an automated vehicle directly using the corner positions and orientation of the slot, but also to work

as input data for slot-pattern recognition in the next section. The parking-line-segment detection stage consists of four steps: Pre-processing for the around-view image, line-marking extraction using the cone-hat filter, marking clustering at a line segment level using D-DBSCAN, and line fitting using RANSAC.

4.2.1 Pre-process for setting Around View Monitoring System

AVM is a system that provides a surround-top-view image by concurrently merging four images from front, left, right, and rear fish-eye cameras. By calibrating the intrinsic and extrinsic parameters of an individual camera and optimizing the distortions and mismatches of the overall camera, an appropriate AVM image can be generated [84]. In this paper, we utilize AVM images with pre-defined calibration parameters and do not directly access individual cameras.

4.2.2 Road Marking Feature Extraction

In parking-slot recognition using parking-line detection, a robust marking extraction filter is a prerequisite for finding line segments. Line-marking features on roads are extracted by considering their ridge properties. The utilization of a ridge filter is able to cover most of the harsh conditions of shadows, dirt, or cracks on the road. As in [82], we extract lane-markings around each pixel by the following equation:

$$\begin{aligned} L[x] &= I[x] * f[x] \\ (I * f)[n] &= \sum^{img.width} I[n-m]f[m] \end{aligned} \tag{4.1}$$

where $I[x]$, $f[x]$ and $L[x]$ indicate image intensity, top-hat filter and its response function respectively. The size of filter $f[x]$ is designed to fit into the marking width. We hold local maxima of filter response $L[x]$ exceeding a certain threshold and regard it as a marking feature. The filter can be built in two-dimensional manner or it can be filtered with respect to x and y axis independently. For all extracted marking features $x_i, \forall i \in \mathcal{N}$, their center position is a utilized dimension for clustering and put to

feature data $\mathbf{F}$, $x_i \in \mathbf{F}$.

4.2.3 Road Marking Clustering using Directional-DBSCAN

In almost all previous work, line detectors are widely adapted for finding line segments, such as the Hough transform or Radon transform, but are fragile for both distorted AVM images and short-length lines, which are seen in most AVM images in parking scenarios. Therefore, we adapt a clustering method rather than utilize the popular line-segment detection algorithm. We propose a novel line-segment clustering algorithm, D-DBSCAN, which basically follows DBSCAN's properties, but clusters within the line-segment level. In this section, we explain the additional suggested feature of D-DBSCAN in comparison with DBSCAN.

DBSCAN

Among the existing clustering algorithms, we have chosen DBSCAN algorithm since it has the ability in discovering clusters with arbitrary shapes such as linear, concave, oval, etc. Furthermore, in contrast to some clustering algorithms, it does not require the predetermination of the number of clusters. DBSCAN [63] has been proven its ability of processing very large databases. Before the mentioning DBSCAN algorithm, a few definitions are explained as:

- Eps: a stretched region to consider as a same cluster.
- MinPts: MinPts stands for minimum points and discriminates whether the point has enough density.
- Core point: A number of a point and its eps-neighbor points is larger than MinPts.
- Border point: A point is eps-neighbor of the core point but its eps-neighborhood does not exceed MinPts.

Algorithm 1 DBSCAN[63]

DBSCAN(D, Eps, MinPts)

```
1: cID = 1
2: for: i ← 1 to |D|
3:     if D[i].cID = Unclassified
4:     if ExpandCluster(D,D[i],cID,Eps,MinPts)=true
5:         cID = cID + 1
6:      end if
7:    end if
8: end for
```

- Noise : A point is the initial seed point and its eps-neighborhood points are lesser than MinPts.
- ClusterID (cID) : A class index of a cluster-set.

DBSCAN determines the density by the number of eps-neighborhood points. Points with a density above a specified threshold are constructed as clusters. By examining all eps-neighorhood of core points in an aforementioned manner, density reachable and connectivity of the cluster is analyzed.

Main algorithm of DBSCAN is presented in Algorithm 1. For each point in data **D**, a undesignated point is examined as a core point based on density within the eps and expanded the examination to eps-neighbor points. This procedure is performed by *ExpandCluster* function. If the cluster meets border points and *ExpandCluster* terminates, a new cluster with the consecutive ClusterID is assigned to unclassified points.

Directional-DBSCAN

The main addition of D-DBSCAN is orientation analysis of core point and eps-neighborhood when expanding adjacent points into the same cluster. In D-DBSCAN, a new property, orientation similarity, is included in addition to original properties (density reachability and density connectivity) of DBSCAN. With original properties in DBSCAN, let a point p be inside of the Eps-neighborhood of a point q and their number $|N_{eps}(q)|$ exceeds $MinPts$. Then, the additional definition for D-DBSCAN is elaborated in the following.

Definition 1: (Orientation Similarity) Let O_f be an orientation of an initial core seed and O_{q2p} be an orientation from q to p, then orientations between O_f and O_{q2p} should be similar within a predefined similarity bound.

The main algorithm flow of D-DBSCAN is the same in DBSCAN(Algorithm 1), but the directivity analysis and orientation similarity evaluation (pseudo-code #6-8, 24-25 and 27 in Algorithm 2) are added from the original *ExpandCluster()* function. When an initial core point and its eps-neighborhood point set is found, a derived orientation by *DirectivityAnalysis()* function among the initial seeds is evaluated whether it is unidirectional right or not. When the orientation is considered as unidirectional, D-DBSCAN continues to find a new point which satisfies both density connectivity and orientation similarity until it meets a border point.

Directivity Analysis

When an initial point which is assigned to the function *ExpandCluster()* and evaluated as a core point, *DirectivityAnalysis()* function begins. The function is to analyze whether the seed points have the sole distinct direction. For achieve this goal, we utilize linear-RANSAC to evaluate the direction of the seed point set. If the iteration during RANSAC among a seed set does not converge at predefined occurrence (e.g., 5) or the inlier ratio does not meet a criterion (e.g., 0.87), the set is considered as omnidirectional and *ExpandCluster()* session is terminated and leave the core point unclassified.

Algorithm 2 ExpandCluster for Directional-DBSCAN

ExpandCluster(D, p, cID, Eps, MinPts)

```
1: seeds ← Neighborhoods(p,Eps)
2: if |seeds| < MinPts
3:     p.cID ← Noise
4:     return false
5: end if
6:  distribute ← ComponentAnalysis(p,seeds,orient)
6: if distribute=directional
7:     attribute ← Directional-DBSCAN
8: else if distribute = congregated
7:     attribute ← DBSCAN
8: else
7:     return false
8: end if
9: p.cID = cID
10: seeds.delete(p)
```

Algorithm 2 ExpandCluster for Directional-DBSCAN

11: ***while***: seeds ! $= \emptyset$
12: q $\leftarrow$ seeds.pop()
13: ***if*** q.cID $\notin$ { *Unclassified, Noise*}
14: continue
15: end ***if***
16: q.cID $\leftarrow$ cID
17: seeds.delete(q)
18: S $\leftarrow$ Neighborhoods(q,Eps)
19: ***if*** |S| $<$ MinPts
20: continue
21: end ***if***
22: ***for***: $x \leftarrow$ 1 to |S|
23: ***if*** x.cID $\notin$ {*Unclassified, Noise*}
24: x.orientation = getOrientation(q.cID, x)
25: ***if*** |x.orientation $-$ orient | $<$ similarity bound
26: seeds.append(x)
27: end ***if***
28: end ***if***
29: end ***for***
30: end ***while***
31: return ***true***

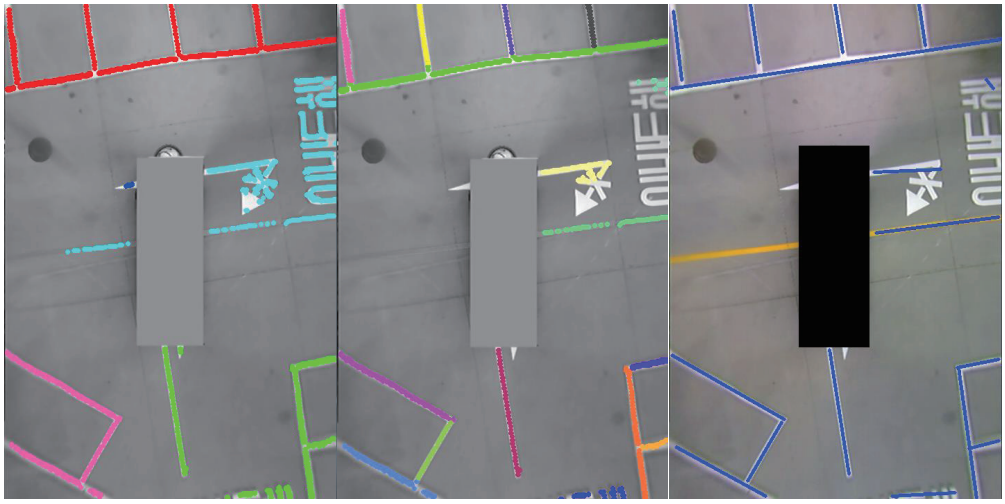


Figure 4.2. From left to right: DBSCAN clustering, D-DBSCAN clustering, line fitting results of D-DBSCAN clusters.

Otherwise, the seed set is assigned as one distinct D-DBSCAN cluster and the converged linear model parameter is converted into *orientation* for checking orientation similarity afterwards.

4.2.4 Line fitting of Line Marking Clusters

Although line-segments are individually clustered with D-DBSCAN, these are only a set of feature points. For deriving more meaningful information such as line information, individual line-segment is fitted with RANSAC based least mean square error fitting. By the Sequential-RANSAC [22], the orientation and intercept of line-segments are collected.

4.3 Slot Pattern Recognition and Detection

Since the previously extracted line-segment itself dose not contain any parking-slot information, meaningful information such as target slot entrance, orientation and pattern should be determined. Parking slots have diverse patterns in order to fit into limited

space. Knowledge of the target slot pattern is not mandatory but significantly important for vehicle maneuvering and facilitate slot detection.

This section presents slot pattern recognition and detection using several geometric cues of slot. It is comprised of slot-group partition, classification of slot patterns and slot detection. We assume that all slots linked together have the similar slot patterns by our practical experience.

4.3.1 Pre-Processing for slot-pattern Recognition: Adjacent parking slot grouping

Since AVM searches every direction, diverse slot-groups having different patterns can be positioned at many side. In this manner, we divide each slot-group based on distance and position of each detected line. For division of slot-group, an undirected graph $G = (V, E)$ is constructed when the tip of the lines T and crossing points of lines C are belong to vertices V, $\{S, C\} \subset V$, whereas distance T between vertices edge E, $\{T\} \subset E$. Using the graph, lines are decomposed into each slot-group node-set on a close distance basis.

When a slot group is partitioned, the slot recognition procedure is performed individually. Within the slot group, lines are categorized into major orientation at most two. Other lines, not belonging to the major orientation, are considered as outliers and discarded. Categorized major lines are discriminated between entrance and lateral lines. For discrimination of line state $\mathbf{x} = \{lateral, entrance\}$, the numbers of each line-category N, their average length L and location of the slot-group P are the subset of observation, $\mathbf{z} = \{N, L, P\}$. With a maximum likelihood function $argmax P(\mathbf{z}|\mathbf{x}) = P(\mathbf{x}|\mathbf{z})P(\mathbf{z})$, we can distinguish the categorized lines between lateral and entrance lines.

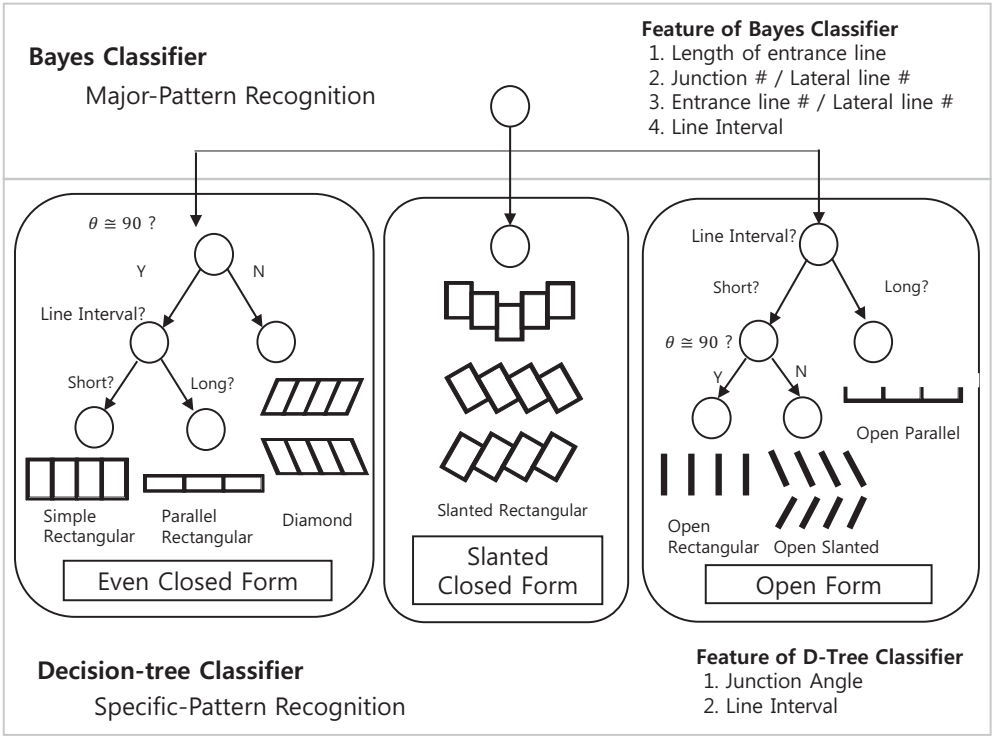


Figure 4.3. Two-step classifier

4.3.2 Slot Pattern Classifier

For slot pattern recognition, we design a two-step classifier to elicit specific-pattern information. The two-step classifier consists of bayes classifier and decision tree. Bayes classifier recognizes a major part of the slot-type among even-closed, slanted-closed or open form. Once the major pattern is classified, decision tree classifier determines its specific pattern. We additionally supplement the some slot patterns slanted open and parallel open to slot-type definition from the study [71]. Fig. 4.3 describes overall structures of two-step classifier for slot patterns.

Bayes Classifier

As evidences of slot pattern recognition, we utilize geometric characteristics from the partitioned slot group individually. The utilized characteristics consist of intercept dis-

tribution of entrance lines, length of entrance line, the number of entrance lines and junctions. Because the number of lines and junctions varies with number of slots, we illuminate the ratio of junction number and entrance line number to lateral line number. For instance, if the larger ratio of entrance line and junction number to lateral lines has, the pattern is more likely to be slanted-closed form. Whereas, if the longer a length of entrance line is, the greater probability on even-closed form is.

Bayesian Network(BN) model is employed for integrating these evidences. With a form of a directed acyclic graph $G = (V, E)$, a node implies that each random variable containing an observable factor and an edge represents a conditional probability of a parent node in BN[20]. In our model, observation nodes $\Im = \{L, J, E, I\}$ contains length of entrance line L, ratio of junctions J and entrance lines E to lateral lines, and intercept distribution I. All observation nodes are assumed to be conditionally independent. By integrating the observations, a prior, slot-pattern s, is classified among $even\ closed$, $slanted\ closed$ and $open$ form. With the BN definition and parameters, the proposed model are shown as:

$$P(s, \Im) = P(s)P(L|s)P(J|s)P(E|s)P(I|s). \tag{4.2}$$

In order to meet our criterion, all parameters, including the probability of edges and a prior, should be trained to perform as prior knowledge. These parameters are determined by solving the maximization problem of the log-likelihood function with previously acquired training data $\mathbf{D}$ containing individual observation $\mathbf{x} = \{s, L, J, E, I\}$.

$$maximize\ \log P(\mathbf{D}|\Theta) = \sum_{\mathbf{x}\in\mathbf{D}} \log P(\mathbf{x}) + \sum_{\mathbf{x}\in\mathbf{D}} \log \sum_{s\in S} P(\Im|s)P(s|\mathbf{x}) \tag{4.3}$$

Decision Tree

Decision tree classifier discriminates a specific part of pattern whether the slot is simple, parellel or inclined within the classified major-pattern. The utilized features for classifying specific patterns are comprised of average of lateral line interval and junction angles. In order to select the best attributes for classifying training dataset, we

adapt Gini-index[83]. As a result shown in fig. 4.3, each attribute is partitioned into two lists and determines the specific slot-pattern properly. Pruning phase for inspecting imperfection and prevents over-fitting does not applied because our model is rather concise.

4.3.3 Parking-slot Detection

When the slot pattern recognition step ends, slot-pattern, junctions, entrance and lateral lines are attained as by-products. We utilize these information to detect parking-slots. Among them, slot junctions are the most important clue for detecting slots. If the major-slot pattern is the open form and no junction exists, tip of the lines are allocated to junctions in open form.

Junction-set are tracked on the consecutive frames using rotation and translation matrix. We assume that AVM image represents the plat ground plane in the real world with correct calibration having only scale difference. Based on the changes in matched lines and junctions of consecutive frames, we can derive rotation θ and translation $\mathbf{t}$ as follows:

$$\mathbf{x}_{i+1} = \mathbf{R}\mathbf{x}_i + \mathbf{t} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}. \tag{4.4}$$

By estimating $\mathbf{x}_{i+1}$ using $\mathbf{R}$ and $\mathbf{t}$, the junction positions are preserved even on certain missed junctions due to occlusions or faint lines.

In this tracked junction-set $\mathbf{J}$, every two-junction pair $j_a \in \mathbf{J}, j_b \in \mathbf{J}$ is evaluated whether they compose a slot entrance. When two conditions(i.e. the distance and orientation of the junction pair) are met, the junction-pair is considered as a parking-slot. These two conditions follow the predefined range corresponding to the recognized slot-pattern.

4.4 Experiments and Evaluation

4.4.1 Experimental Set-up

The experiments were conducted with an autonomous vehicle platform and the cameras are mounted around the vehicle to form an AVM image. An AVM image sequence is directly obtained in 50fps with an ECU board for the process. The algorithm is implemented by C++/OpenCV and runs at a frame rate of 31.7 fps on a normal PC(Intel Core i5 Q720 CPU @ 2.2GHz and 8.00GB RAM) with an image size of 480x720. This includes all procedures except for image acquisition. To achieve effective line-level clustering using D-DBSCAN, applied parameters are as follows: eps =29.3px, MinPts=6EA, orientation similarity bound=1.6°, when marking features are extracted per 3px.

We have acquired dataset containing diverse parking patterns and environments. Simple and slanted rectangular, diamond, open-slanted, open-simple types are included. As a dataset, 3140 frames used for the recognition training set and 1730 frames are tested for parking slot detection evaluation. We adapt two classic measures for performance evaluation: precision = TP/(TP+FP) and recall = TP/(TP+FN), when TP, FN and FP are abbreviation of true positive, false negative and false positive.

4.4.2 Evaluation

The contributions of this study are illuminated by the evaluation of two factors: 1) a performance of the proposed Directional-DBSCAN algorithm and comparison to other feature clustering methods, 2) an advancement of road environment perception algorithms using the proposed clustering algorithm.

Evaluation of Directional-DBSCAN Algorithm

We evaluate instance-wise clustering performance of Directional-DBSCAN among extracted marking feature points from acquired real AVM data. We manually labeled the

Table 4.2. Performance evaluation in line-segment clustering.

Cases	Points # (Line segments #)	Recall	Precision
Sequence1	1394 (8)	0.957	0.915
Sequence2	1010 (10)	0.948	0.947
Sequence3	1504 (13)	0.978	0.943
Sequence4	1616 (11)	0.934	0.903

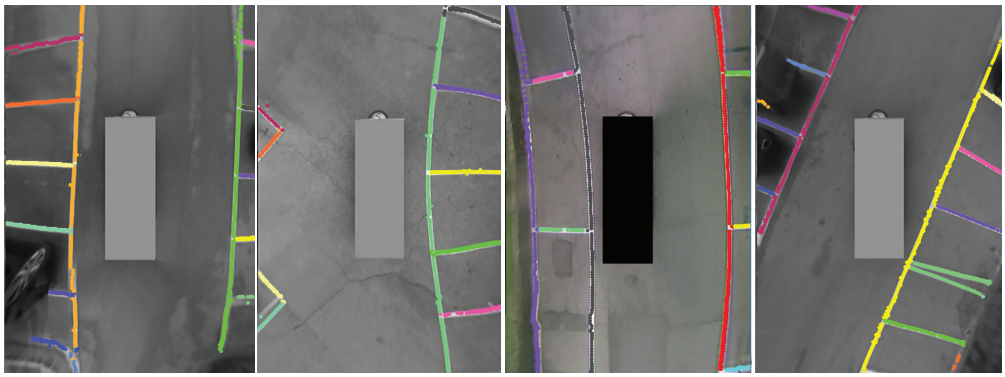


Figure 4.4. Directional-DBSCAN clustering results in the dataset.

road marking points with each line segment level for generating ground truth set. Metrics are defined as follow,

$$recall = \frac{number\ of\ correctly\ clustered\ feature\ points}{number\ of\ existing\ feature\ points}, \tag{4.5}$$

$$precision = \frac{number\ of\ correctly\ clustered\ feature\ points}{number\ of\ clustered\ feature\ points}. \tag{4.6}$$

The following fig. 4.4 is the result of clustering the marking points in each line instance. The performance of using this dataset has been evaluated in the Table 4.2.

Table 4.3. Performance comparison between different modules for its detection purpose.

	Line-based PSD		Corner-based PSD		Lane Detection	
	Prob. HT[65]	Ours	Fast[68]	Ours	DBSCAN[87]	Ours
Recall	0.872	0.951	0.826	0.943	0.931	0.935
Precision	0.926	0.918	0.937	0.962	0.826	0.953

Evaluation of Instance-level Detection Algorithm: Lane, parking line and corner

We compare the proposed D-DBSCAN algorithm is compared other instance detectors. For evaluation, recall and precision are utilized detection metrics as below,

$$recall = \frac{number\ of\ correctly\ detected\ instances}{number\ of\ existing\ instances}, \tag{4.7}$$

$$precision = \frac{number\ of\ correctly\ detected\ instances}{number\ of\ detected\ instances}. \tag{4.8}$$

For parking slot detection, probabilistic Hough transform and fast corner detectors. Targets consist of 2338 lines and 2018 corners from considerably warped unflat region dataset. For comparison to corner detector, we compare extracted crossing points from D-DBSCAN line-segments. Fig. 4.8 and Table 4.3 show our proposed algorithm is robust on detecting short or distorted line-segments. In addition, line-detection-based algorithm has higher performance even in a single frame. Corner detector based algorithms are fragile on the occluded or faint corners of far-reached regions.

On the other hand, the lane detection performance comparison using other instance-clustering methods are performed. For instance clustering, DBSCAN algorithm [87] is compared for lane marking clustering. The algorithms are evaluated in various types of lane shapes such as normal, merging, splitting, curvy and double lanes. The proposed method has merits in clustering various types of aforementioned lane marking scenarios. The clustering performance is shown in the right part of Table 4.3. Among them,

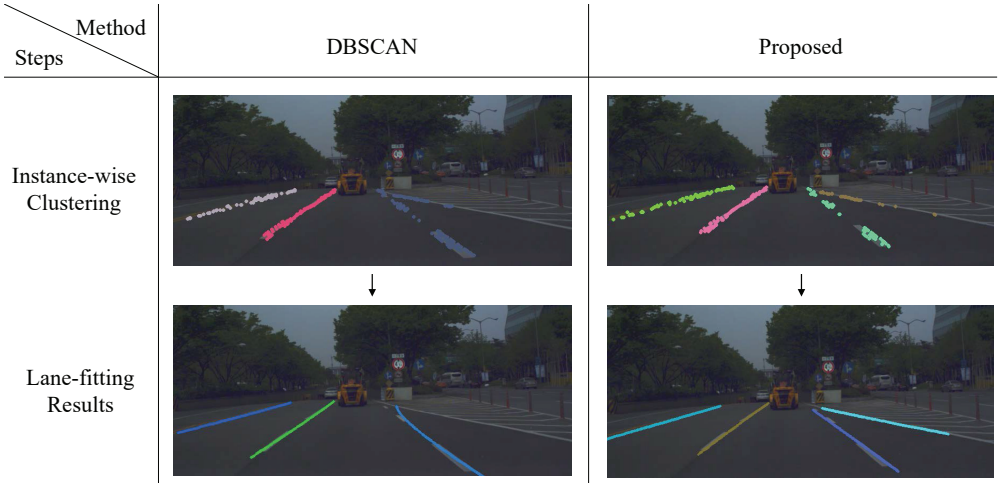


Figure 4.5. Lane marking clustering of feature points using Directional-DBSCAN.

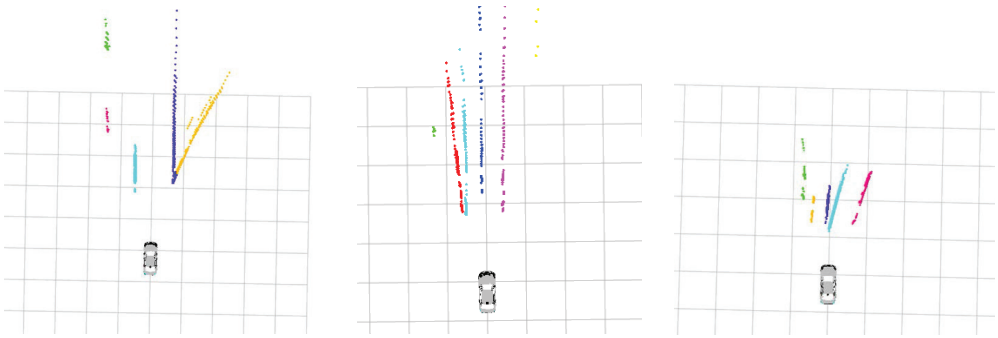


Figure 4.6. Example of proper lane-instance level clustering in top view perspective.

Fig. 4.5 displays an obvious example of merit of a merging lane scenario. In this figure, original DBSCAN algorithm cluster all spatial connected points and merged lane points are clustered within a single cluster. Therefore, the lane fitting of the improper clustered set is in trouble to fit properly and only one lane is detected with slight lane coefficient error. Meanwhile, our proposed method detach two merging lanes within two lane properly and it eventually detects two lanes properly. Cases of clustering various merging and splitting lane are shown in Fig. 4.6 and proper fitting results are shown in Fig. 4.7.

Figure 4.7. Example of proper lane-instance level clustering in front view perspective.

Table 4.4. Slot-detection performances on diverse environments.

Scenarios	# of test slots	Recall	Precision
Sunny	4794	0.962	0.915
Underground	2210	0.958	0.897
Sunset	1104	0.948	0.923
Rainy	816	0.924	0.903
Snowy	742	0.896	0.920

Evaluation of Parking slot detection

Moreover, we evaluate parking-slot detection performance with varying scenarios. We evaluate with different types of environments including sunny, sunset, rainy and underground. Since our algorithm targets a parking slot with two existing junction, we only consider a slot target to ground truth when it is a parking slot with two visible junctions. As shown in test results at Fig. 4.8, we include various types of parking slots such as simple rectangular, diamond, open, open slanted, parallel and slanted rectangular slots patterns for testing set. Moreover, we evaluate the performance of parking-slot detection algorithm on different slot environments as shown in Fig. 4.9. The comparison result is shown at Table 4.4. This indicates that the proposed algorithm shows decent performances even in considerably tough condition such as rainy or underground regions with lots of light-reflections.

Fig. 4.10 shows importance of parking grouping. Even if there are difficult en-

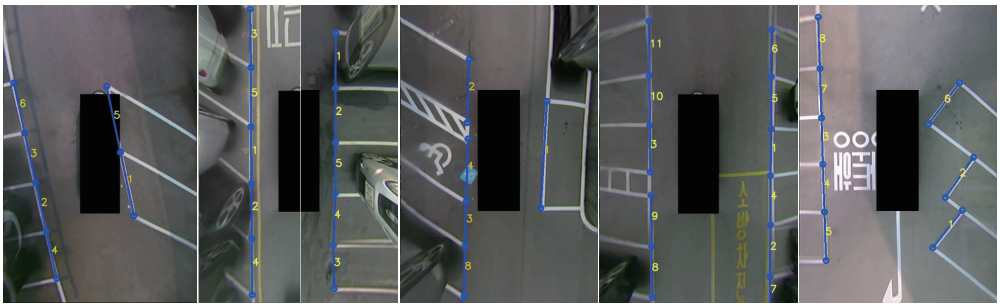


Figure 4.8. Test results in diverse types of slots(simple rectangular/diamond(1st), diamond with double lines/open(2nd), open slanted/parallel(3rd), rectangular distortion on unflat terrains(4th), slanted rectangular(5th).

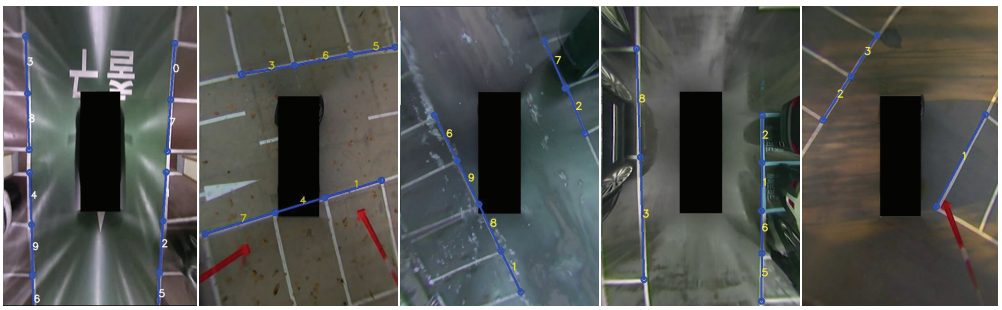


Figure 4.9. Test results in diverse environments scenarios (underground, fallen leaves, snowy, rainy and sunset).

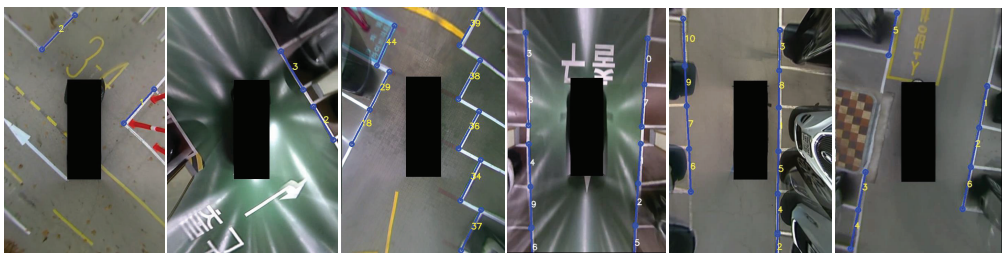


Figure 4.10. Test results in various difficult environments (co-existence of pillar, vehicle occlusion, brick lines, curb and irrelevant types of road markers.)

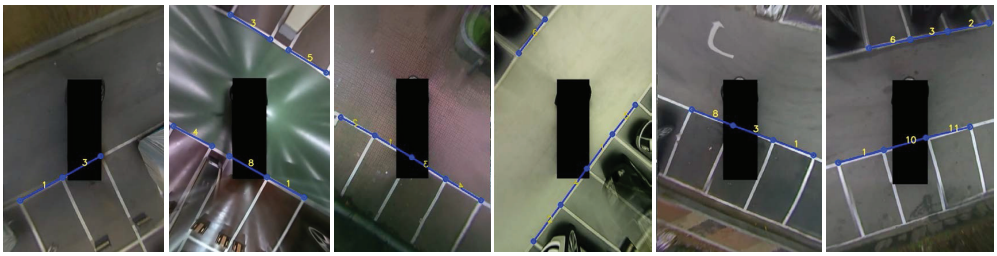


Figure 4.11. Detection of parking moment after selecting parking space.

vironments such as co-existence between slot and irrelevant outliers such as a white pillar, white vehicle occlusion, brick lines, curb, lanes and road markers, slot-grouping avoids the road outliers. Meanwhile, Fig. 4.10 presents the moment of vehicle parking when the parking target is chosen. Because of the slot junction tracking, the target slots keeps detected even when the ego-vehicle occlude any junction of the target parking slot.

Evaluation of Localization Enhancement

In order to show importance of feature clustering, enhancement of localization accuracy from different lane detector is evaluated. Localization accuracy is evaluated within two methods: lane map matching using DBSCAN-based lane detection [87] and lane map matching using Directional-DBSCAN-based lane detection results. The same map matching model is adpated for performance evaluation in order to exclude other elements and elaborate the effects of lane clustering enhancement. The utilized localization method is the particle filter-based map matching methods such like the previous chapters with 1000 pose candidates. An initial pose is assumed given value. Tested trajectory, 1.5km in total, is partial region from dataset B in chapter 3 as shown in Figure 4.12.

As shown in Table 4.5, the evaluated localization performance is enhanced after the lane detection performance is improved with Directional-DBSCAN. Target evaluation is confined to lateral pose error and orientation error because our proposed method

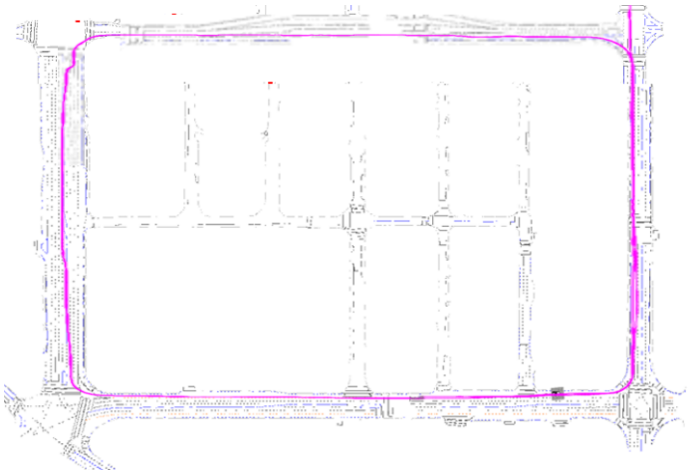


Figure 4.12. Localization result in the target zone.

mainly enhances lane detection results. Since detection performance of merged and split lane are enhanced, the improved lane measurement affect enhancing localization error correspondingly. This proves that robust road marker feature association algorithm is inevitable not only in local positioning such as parking slot detection or lane detection, but also in map matching based localization methods.

Table 4.5. Localization performances: Mean (Standard dev.)

Scenario	Dataset B from Ch.3	
Accuracy / Methods	Lateral(m)	Orientation(°)
Matching using lanes with DBSCAN [87]	0.332 (0.42)	0.57 (0.29)
Matching using lanes with proposed method	**0.237** (0.23)	**0.55** (0.30)

4.5 Conclusion

This paper proposes a parking-slot detection algorithm based on D-DBSCAN algorithm for line-segment detection. Through quantitative analysis, we evaluate the D-DBSCAN algorithm outperforms on detecting blurry and short line-segments in AVM images. Considering that most of the AVM images contain multiple short and blurry parking-lines frequently, our algorithm is more suitable on AVM-based systems. We plan to develop our study to build AVM-based-map and localization accordingly.

Chapter 5

Conclusion

In this dissertation, we have proposed road semantic perception and localization algorithms that can be used in automated driving vehicles. Since we are inspired by the benefits of road semantic map matching methods in lateral localization accuracy, novel algorithms are suggested to improve the overall map matching performance. Directional-DBSCAN suggested in chapter 4 improves the road markings and lane detection performance of with efficient data association within road-line instance. This algorithm enhances lane detection performance dramatically especially in merged or split lanes.

At the same time, identified ego-lane index information proposed in chapter 3 plays an important role in semantic map matching since it prevents confusion to adjacent lanes. In order to classify the stable ego-lane index even in harsh urban conditions, LSTM layer is employed to analyse in a sequential manner after lane detection network is modified to extract various lane type recognition. Eventually, road network map matching can avoid matching errors in other lanes.

This improved road network map matching source is placed complementary to point cloud map matching. Since the two types of features from map matching sources are then separated, the integration of this heterogeneous matching source can ensure various fail-safe scenarios and this assures autonomous driving safety in localization

aspect. After scoring sensor and local map's feature sufficiency, the map matching quality is evaluated. The importance of certain matching source is derived after feature sufficient score is combined by DST framework. Therefore, the complementary integration of these two disparate map matching source allows the overall localization system to be performed in a fail-safe manner.

Bibliography

[1] Y. Jiang and F. Gao, "Computer vision-based multiple-lane detection on straight road and in a curve," *Image Analysis and Signal Processing*, Zhejiang, China, 2010.

[2] M. Nieto and L. Salgado, "Robust multiple lane road modeling based on perspective analysis," *Image Processing*, ICIP, San Diego, 2008.

[3] S. Kang, S. Lee, J. Hur and S. Seo, "Multi-lane Detection based on Accurate Geometric Lane Estimation in Highway Scenarios," in *Intelligent Vehicles Symposium(IV)*, Dearborn, Michigan, 2014.

[4] V. Popescu, S.Nedevschi, "Lane Identification and Ego-vehicle Accurate Global Positioning in Intersections," in *Proc. IEEE Conf. Intelligent Transportation Systems*, Baden-baden, Germany, 2011.

[5] S. Nedevschi, F.Oniga, "Accurate Ego-Vehicle Global Localization at Intersections Through Alignment of Visual Data With Digital Map," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 14, No.2, June, 2013.

[6] K.Choi, J.Park, "Methods to Detect Road Features for Video-Based In-Vehicle Navigation System," *Journal of Intelligent Transportation Systems*, Vol 14. No.1, 2010.

[7] T. Kuhnl, J.Fritsch, “Visual Ego-Vehicle Lane Assignment using Spatial Ray Features,” in *Proc. Intelligent Vehicles Symposium*, Columbus, Gold Coast, Australia, 2013.

[8] Y. Seo, R.Rajkumar, “Tracking and Estimation of Ego-Vehicle’s State for Lateral Localization,” in *IEEE 17th International Conference on Intelligent Transportation Systems(ITSC)*, Qingdao, China, 2014.

[9] A. Hata, D. Wolf, “Road Marking Detection Using LIDAR Reflective Intensity Data and its Application to Vehicle Localization,” in *IEEE Intelligent Vehicles Symposium(IV)*, Dearborn, Michigan, 2014.

[10] J.M. Alvarez, A.M. Lopez, “Road Detection Based on Illumination Invariance,” *IEEE Transactions on Intelligent Transportation Systems*, Vol. 12, No.1, March, 2013.

[11] J. Hur, S. Seo. “Multi-Lane Detection in Highway and Urban Driving Environment.” in *Master Thesis* at Seoul National University, 2013.

[12] J. Hur, S. Kang and S. Seo. “Multi-lane detection in urban driving environments using conditional random fields.” in IEEE Intelligent Vehicles Symposium, Gold Coast, Austrailia, 2013

[13] D. Barber, Bayesian Reasoning and Machine Learning, *Cambridge University Press*, 2013, ch.6.

[14] M. Nieto and L. Salgado, “Robust multiple lane road modeling based on perspective analysis,” *ICIP*, San Diego, 2008.

[15] A. Geiger, M. Lauer, C. Wojek, C. Stiller and R. Urtasun, “3D Traffic Scene Understanding From Movable Platforms,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 36, No.5, 2014

[16] Yu, Fisher, et al. “BDD100K: A diverse driving video database with scalable annotation tooling.” arXiv preprint arXiv:1805.04687 (2018).

[17] Liu, Guoliang, Florentin Wörgötter, and Irene Markelić. “Combining statistical hough transform and particle filter for robust lane detection and tracking.” 2010 IEEE Intelligent Vehicles Symposium. IEEE, 2010.

[18] Xie, Saining, and Zhuowen Tu. “Holistically-nested edge detection.” Proceedings of the IEEE international conference on computer vision. 2015.

[19] Simonyan, Karen, and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:1409.1556 (2014).

[20] Lee, Soomok, Seong-Woo Kim, and Seung-Woo Seo. “Accurate ego-lane recognition utilizing multiple road characteristics in a Bayesian network framework.” Intelligent Vehicles Symposium (IV), 2015 IEEE. IEEE, 2015.

[21] Lee, Soomok, et al. “Directional-DBSCAN: Parking-slot detection using a clustering method in around-view monitoring system.” Intelligent Vehicles Symposium (IV), 2016 IEEE. IEEE, 2016.

[22] Lee, Soomok, and Seung-Woo Seo. “Available parking slot recognition based on slot context analysis.” IET Intelligent Transport Systems 10.9 (2016): 594-604.

[23] Du, Xinxin, and Kok Kiong Tan. “Comprehensive and practical vision system for self-driving vehicle lane-level localization.” IEEE transactions on image processing 25.5 (2016): 2075-2088.

[24] Cui, Dixiao, Jianru Xue, and Nanning Zheng. “Real-time global localization of robotic cars in lane level via lane marking detection and shape registration.” IEEE Transactions on Intelligent Transportation Systems 17.4 (2016): 1039-1050.

[25] Vivacqua, Rafael Peixoto Derenzi, et al. “Self-localization based on visual lane marking maps: An accurate low-cost approach for autonomous driving.” IEEE Transactions on Intelligent Transportation Systems 19.2 (2018): 582-597.

[26] Jang, Eun, Jae Suhr, and Ho Jung. “Lane endpoint detection and position accuracy evaluation for sensor fusion-based vehicle localization on highways.” Sensors 18.12 (2018): 4389.

[27] Guo, Chunzhao, et al. “A low-cost solution for automatic lane-level map generation using conventional in-car sensors.” IEEE Transactions on Intelligent Transportation Systems 17.8 (2016): 2355-2366.

[28] Suhr, Jae Kyu, et al. “Sensor fusion-based low-cost vehicle localization system for complex urban environments.” IEEE Transactions on Intelligent Transportation Systems 18.5 (2017): 1078-1086.

[29] Huval, Brody, et al. “An empirical evaluation of deep learning on highway driving.” arXiv preprint arXiv:1504.01716 (2015).

[30] Neven, Davy, et al. “Towards end-to-end lane detection: an instance segmentation approach.” 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018.

[31] Li, Jun, et al. “Deep neural network for structural prediction and lane detection in traffic scene.” IEEE transactions on neural networks and learning systems 28.3 (2017): 690-703.

[32] Kim, Jihun, and Minho Lee. “Robust lane detection based on convolutional neural network and random sample consensus.” International Conference on Neural Information Processing. Springer, Cham, 2014.

[33] Gopalan, Raghuraman, et al. “A learning approach towards detection and tracking of lane markings.” IEEE Transactions on Intelligent Transportation Systems 13.3 (2012): 1088-1098.

[34] Su, Yingna, et al. “Vanishing point constrained lane detection with a stereo camera.” IEEE Transactions on Intelligent Transportation Systems 19.8 (2018): 2739-2744.

[35] Chen, Chenyi, et al. “Deepdriving: Learning affordance for direct perception in autonomous driving.” Proceedings of the IEEE International Conference on Computer Vision. 2015.

[36] Ballardini, Augusto Luis, et al. “Ego-lane estimation by modeling lanes and sensor failures.” 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2017.

[37] Cheng, Hsu-Yung, et al. “Ego-Lane Position Identification with Event Warning Applications.” IEEE Access (2019).

[38] Rabe, Johannes, Marc Necker, and Christoph Stiller. “Ego-lane estimation for lane-level navigation in urban scenarios.” 2016 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2016.

[39] Hyeon, D., Lee, S., Jung, S., Kim, S. W. and Seo, S. W. “Robust road marking detection using convex grouping method in around-view monitoring system.” In Intelligent Vehicles Symposium (IV), 2016 IEEE (pp. 1004-1009). IEEE.

[40] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks.” Advances in neural information processing systems. 2012.

[41] Geiger, Andreas, et al. “Vision meets robotics: The KITTI dataset.” The International Journal of Robotics Research 32.11 (2013): 1231-1237.

[42] Kingma, Diederik P., and Jimmy Ba. “Adam: A method for stochastic optimization.” arXiv preprint arXiv:1412.6980 (2014).

[43] Zhao, Hengshuang, et al. “Icnet for real-time semantic segmentation on high-resolution images.” Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[44] Howard, Andrew G., et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications.” arXiv preprint arXiv:1704.04861 (2017).

[45] Forman, George. “An extensive empirical study of feature selection metrics for text classification.” Journal of machine learning research 3.Mar (2003): 1289-1305.

[46] Grisetti, G., Kummerle, R., Stachniss, C. and Burgard, W., 2010. A tutorial on graph-based SLAM. IEEE Intelligent Transportation Systems Magazine, 2(4), pp.31-43.

[47] Mur-Artal, R., Montiel, J.M.M. and Tardos, J.D., 2015. “ORB-SLAM: a versatile and accurate monocular SLAM system.” IEEE Transactions on Robotics, 31(5), pp.1147-1163.

[48] Engel, Jakob, Jorg Stückler, and Daniel Cremers. “Large-scale direct SLAM with stereo cameras.” Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. IEEE, 2015.

[49] Levinson, Jesse, and Sebastian Thrun. “Robust vehicle localization in urban environments using probabilistic maps.” Robotics and Automation (ICRA), 2010 IEEE International Conference on. IEEE, 2010.

[50] Liang Li, Ming Yang, Chunxiang Wang, and Bing Wang, “Hybrid Filtering Framework based Robust Localization for Industrial Vehicles,” IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, 2018 (accepted).

[51] Hyunsung Lee, Seonwook Kim, Sungyoul Park, Yonghwan Jeong, Hojoon Lee, and Kyongsu Yi, “AVM / LIDAR Sensor based Lane Marking Detection Method

for Automated Driving on Complex Urban Roads," Intelligent Vehicles Symposium (IV), 2015 IEEE. IEEE, 2017.

[52] Gyu-Min Oh and Seung-Woo Seo, "Fast Localization for Autonomous Driving based on Normal Distributions Transform," 2018 International Conference on Electronics, Information, and Communication, 2018.

[53] Wan, Guowei, et al. "Robust and Precise Vehicle Localization based on Multi-sensor Fusion in Diverse City Scenes." arXiv preprint arXiv:1711.05805 (2017).

[54] De Silva, Varuna, Jamie Roche, and Ahmet Kondoz. "Fusion of LiDAR and Camera Sensor Data for Environment Sensing in Driverless Vehicles." arXiv preprint arXiv:1710.06230 (2017).

[55] Zhang, Ji, and Sanjiv Singh. "Visual-lidar odometry and mapping: Low-drift, robust, and fast." In Robotics and Automation (ICRA), 2015 IEEE International Conference on, pp. 2174-2181. IEEE, 2015.

[56] Zhang, Xinzheng, Ahmad B. Rad, and Yiu-Kwong Wong. "Sensor fusion of monocular cameras and laser rangefinders for line-based simultaneous localization and mapping (SLAM) tasks in autonomous mobile robots." Sensors 12.1 (2012): 429-452.

[57] Jo, Sungjin, Hyukdoo Choi, and Euntai Kim. "Ceiling vision based slam approach using sensor fusion of sonar sensor and monocular camera." Control, Automation and Systems (ICCAS), 2012 12th International Conference on. IEEE, 2012.

[58] Kim, Hyungjin, et al. "Robust vehicle localization using entropy-weighted particle filter-based data fusion of vertical and road intensity information for a large scale urban area." IEEE Robotics and Automation Letters 2.3 (2017): 1518-1524.

[59] Shin, Myung-Ok, et al. “Real-time and accurate segmentation of 3-D point clouds based on Gaussian process regression.” IEEE Transactions on Intelligent Transportation Systems 18.12 (2017): 3363-3377.

[60] Himmelsbach, Michael, Felix V. Hundelshausen, and H-J. Wuensche. “Fast segmentation of 3D point clouds for ground vehicles.” 2010 IEEE Intelligent Vehicles Symposium. IEEE, 2010.

[61] Zhou, Yin, and Oncel Tuzel., “Voxelnet: End-to-end learning for point cloud based 3d object detection.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[62] Zhang, L., Wei, L., Shen, P., Wei, W., Zhu, G., and Song, J., “Semantic SLAM Based on Object Detection and Improved Octomap.” IEEE Access, 6, 75545-75559.

[63] M.Ester *et al*, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” Conf. on Knowledge Discovery and Data Mining, pp. 226-231, Portland, 1996.

[64] Jung, H.G., Kim, D.S., Yoon, P.J., Kim, J., “Parking slot markings recognition for automatic parking assist system”, IEEE Intelligent Vehicles Symposium, pp. 106-113, 2006.

[65] Hamada, K., Hu, Z., Fan, M. and Chen, H., “Surround View based Parking Lot Detection and Tracking,” in IEEE Intelligent Vehicles Symposium, South Korea, 2015.

[66] C.Wang and M.Yang et al. “Automatic parking based on a bird’s eye view vision system,” Adv. in Mec. Eng., vol 2014, 2014, pp 1-13.

[67] I. Shim, J.Choi, S. Shin, D. Shim, I. Kwon, "An Autonomous Driving System for Unknown Environments Using a Unified Map," in IEEE Transactions on Intelligent Transportation Systems.

[68] X. Sevillano et al, "Towards smart traffic management systems: Vacant on-street parking spot detection based on video analytics," 17th International Conference on Information Fusion, Salamanca, July 2014

[69] Suhr, J.K., Jung, H. G., "Fully-automatic recognition of various parking slot markings in Around View Monitor (AVM) image sequences," IEEE Conference on Intelligent Transportation Systems, Alaska, 2012.

[70] Suhr, J. K, Jung, H. G., "Full-automatic recognition of various parking slot markings using a hierarchical tree structure," Optical Engineering 52(3),037203, 2013.

[71] Suhr, J.K. Jung, H.G., "Sensor Fusion-based Vacant Parking Slot Detection and Tracking," Transactions on Vehicular Technology, 15(1), 21-36, Feb. 2014.

[72] Choi, J., Chang, E., Yoon, D., Ryu, S., et al, "Sensor Fusion-Based Parking Assist System," SAE Technical Paper 2014-01-0327, 2014.

[73] Jung, H.G., Lee, Y.H., "Uniform User Interface for Semi-automatic Parking Slot Markings Recognition," Transactions on Vehicular Technology, 59(2), 616-626, Feb. 2010.

[74] P. Degerman *et al*, "Hough transform for parking space estimation using long range ultrasonic sensors," SAE Technical Paper, 2006.

[75] Satonaka, H., Okuda, M., Hayasaka, S., Endo, T., Tanaka, Y., Yoshida, T., "Development of parking space detection using an ultrasonic sensor." The 13th World Congress on Intelligent tansportation Systems and Services, 2006.

[76] Gorner, S., Rohling, H., "Parking lot detection with 24 GHz radar sensor", The 3rd International Workshop on Intelligent Transportation, 2006.

[77] H.G Jung, Y. H. Cho, P. Yoon, J. Kim "Scanning Laser Radar-Based Target Position Designation for Parking Aid System," IEEE Transactions on Intelligent Transportation Systems, Vol. 9, No. 3, 2008

[78] Suhr, J. K., Jung, H. G., Bae, K., and Kim, J, "Automatic free parking space detection by using motion stereo-based 3D reconstruction," Machine Vision and Applications, 21(2), pp. 163-176. 2011.

[79] D. Llorca, S.Alvarez, "Vision-based parking assistance system for leaving perpendicular and angle parking lots." in IEEE Intelligent Vehicles Symposium, Austrailia, 2013

[80] E. Pollard, D. Gruyer, A. Cord, "Lane Marking Extraction with Combination Strategy and Comparative Evaluation on Synthetic and Camera Images," IEEE Conference on Intelligent Transportation Systems, Washington DC, USA. pp. 1741-1746, October 2011.

[81] Su, B. and Lu, S., "A System for Parking Lot Marking Detection," pp. 268-273, vol. 8879, Advances in Multimedia Information Processing-PCM, Springer, 2014

[82] M. Nieto and L. Salgado, "Robust multiple lane road modeling based on perspective analysis," *Image Processing*, ICIP, San Diego, 2008.

[83] S. Ranka *et al*, "CLOUDS: A decision tree classifier for large datasets," Conf. on Knowledge Discovery and Data Mining, pp. 1-8, 1998.

[84] K. Sung *et al*, "Development of Image Synthesis Algorithm with Multi-Camera," IEEE Conf. on Vehicular Technology Conference, Yokohama, 2012.

[85] Chavez-Garcia, Ricardo Omar, and Olivier Aycard. "Multiple sensor fusion and classification for moving object detection and tracking." IEEE Transactions on Intelligent Transportation Systems 17.2 (2016): 525-534.

[86] Jo, Kichun, et al. “Precise localization of an autonomous car based on probabilistic noise models of road surface marker features using multiple cameras.” IEEE Transactions on Intelligent Transportation Systems 16.6 (2015): 3377-3392.

[87] Lee, Seokju, et al. “Vpgnet: Vanishing point guided network for lane and road marking detection and recognition.” Proceedings of the IEEE International Conference on Computer Vision. 2017.

초 록

자율주행 기술은 도로 운송 분야에서 더 많은 사용자를 넓힐 수 있는 기회를 제공할 뿐만 아니라 인류에게 새로운 생활 패턴의 패러다임을 제시하는 중요한 연구 주제이다. 이 기술을 통해 교통 사고를 줄이고 상당한 교통 체증을 해결할 수 있다는 점에서 사회적 문제 또한 해결 할 수 있다. 자율주행을 실현하기 위해 환경인지 센서, IMU, GPS 등 다양한 종류의 센서를 결합한 여러 현지화 방법이 제안되어 왔다. 그 중, GPS가 전통적 위치 인식에서 상당한 부분을 차지해 왔으나, 복잡한 도심환경에서는 대부분의 GPS 시스템이 빈번하게 실제 위치에서 더 멀리 떨어진 부정확한 위치를 표기한다. 때문에 GPS의 한계를 보정하기 위해서 환경센서와 해당 센서 관련 정밀 지도를 활용한 측위 방식들이 부각 되었다.

이러한 연구의 주요 관심사는 시스템의 정확성과 정밀도를 극대화하는 동시에 비용을 최소화하는 것이다. 그러나 이들 연구의 대부분은 단일 순간의 측위 성능에 집중한 나머지, 측위 기술의 안정성을 크게 고려하지 않았다. 따라서 본 논문에서는 정확하고 비용 효율적인 측위를 기본으로, 차량 본위의 안정성(fail-safe) 측면을 강조한다. 최대한의 측위 안정성을 확보하는 여러 방법을 제시함으로써 자율주행 측위 분야에 기여하고자 한다. 이 목표를 달성하기 위해, 안정화된 도로 노면에 대한 카메라기반 환경인식 알고리즘, 인식 결과와 도로망 지도의 매칭을 활용한 강건한 측위 기법, 측위 매칭소스의 다중화를 통한 상호보완적 시스템 구성을 바탕으로 최대의 안정성을 보장하는 프레임워크를 제안했다.

우선, 자차 차로 인지기반 측위 시스템을 제안한다. 단일 통합 심층 학습 네트워크에서 다중 차선 검출 및 자차 차로 식별 바탕으로 실시간 처리 조건의 보증 하에

측위 정확성을 보장한다. 인식된 차선 지수 정보가 차로인지 기반 측위 시스템의 안정성에 미치는 영향은 다양한 악의적인 환경에서의 실험을 바탕으로 입증된다.

또한, 차선 기반 측위 시스템은 다른 지도 매칭 기반 측위 시스템과 융합되어 전반적인 측위 안정성을 추구하기 위한 방안으로 확장이 가능하다. 이러한 목표를 달성하기 위해 독립적인 지도 매칭 소스를 이용한 분산적 측위 프레임워크를 제안한다. 분산된 측위 프레임워크는 두 개의 독립적인 지도 매칭 소스를 활용하고 이를 확률적 상황 분석 모델과 통합한다. 확률적 모델을 채택하면 두 개의 지도 매칭 소스의 신뢰성을 계산하고 신뢰성 기반 시스템 안정성을 통해 fail-safe 문제를 해결한다.

마지막으로, 이러한 측위 알고리즘의 근본적 개선은 도로상 노면 표시 검출 성능의 안정성이 유지되는 것이어야 한다. 제 4장에서는 차선과 주차선을 적절히 이용하기 위한 라인 세그먼트 레벨 클러스터링 방법을 제안한다. 제안된 Directional-DBSCAN 클러스터링 알고리즘은 효과적인 도로 차선, 주차선 등의 선 성분 검출을 수행한다. 제안된 클러스터링 방법의 효과적이고 견고함을 입증하기 위해 주차 슬롯 감지 및 차선 감지를 평가하였다. 병합되거나 분할된 경우에도 도로 선을 강건하게 추출한다는 것을 보여준다.

주요어: 차량 측위, 이미지 처리, 기계 학습, 차선인식, 센서융합
학번: 2013-20850