End-to-End Ego-Vehicle Localization using Multi-sensor and a Low Cost Map

Abderrahim Kasmi

UNIVERSITÉ CLERMONT AUVERGNE
ÉCOLE DOCTORALE
SCIENCES POUR L'INGÉNIEUR DE CLERMONT-FERRAND

THÈSE

Présentée par

ABDERRAHIM KASMI
Master Robotique

pour obtenir le grade de

Docteur d'Université

Spécialité : **Electronique et Systèmes**

End-to-End Ego-Vehicle Localization using Multi-sensor and a Low Cost Map

Soutenue publiquement le 15/07/2021 devant le Jury composé de :

MICHELE ROMBAUT	Rapporteur	Professeur à l'Université Grenoble Alpes
VINCENT FREMONT	Rapporteur	Professeur à l'Ecole Centrale de Nantes
PHILIPPE BONNIFAIT	Président du jury	Professeur à l'Université de Technologie de Compiègne
DIEUMET DENIS	Examinateur	Chef de Projet, Sherpa Engineering
ROMUALD AUFRERE	Examinateur	Maître de conférences - HDR à l'Université Clermont-Auvergne
ROLAND CHAPUIS	Directeur de thèse	Professeur à l'Université Clermont-Auvergne

In the memory of my aunt Fatiha.

ACKNOWLEDGEMENTS

Completing a Ph.D. is a long and difficult path that would be difficult to complete on one's own. I was lucky to meet fantastic people that have made it easier and I'm grateful for that today.

My sincere thanks go to my supervisor Roland Chapuis for giving me the opportunity to work on the rising topic of autonomous vehicles. Your constant support, enthusiasm, and wisdom have been essential for me during these years. I owe a debt of gratitude to Romuald Aufrère, who, for some reason, never failed to make me feel enthused about my work. I'm not sure how you accomplished it, but you figured it out.

This Ph.D. was a collaboration between Institut Pascal and the R&D department of Sherpa Engineering. In this regard, I would like to thank Sherpa Engineering for their trust and their financial support and specially Dieumet Denis for his commitment and implication in the smooth running of this work.

I would also like to thank the members of my jury for thoroughly reading the manuscript, their encouraging words and thoughtful, detailed feedback have been very important to me. I would like to thank Philippe Bonnifait for cordially accepting to be the chairman of the thesis committee, as well as Michèle Rombaut and Vincent Fremont for reviewing this dissertation.

I was lucky to be working with amazing people at my office Laurent, Serge, Cyril, Marwa. A special thank to my office coworker Johann Laconte. Your desire to learn and share energize our office. My thank goes also to Kamel, the coffee breaks with you were always a good place to learn a lot, on lots of topics.

Lastly, my family deserves endless gratitude: my father for teaching me how to be critical (maybe too much), my mother for teaching me to never back down. My brothers for teaching me how to be confident in every situation. Last but not least, I would like to thank my wife Dimia, who has been supporting me during this period, nothing would have been possible without you. To my family, I give everything, including this.

Finally, I end this acknowledgment by citing a famous singer Snoop Dog *"I want to thank me for believing in me, I want to thank me for doing all this hard work. I wanna thank me for having no days off. I wanna thank me for never quitting. I wanna thank me for always been a giver and trying to give more than I receive. I want to thank me for trying to do more right than wrong. I want to thank me for just being me at all times"*

ABSTRACT

Nowadays, Autonomous Vehicles (AVs) are capable of realizing extraordinary and complicated tasks. Notwithstanding these amazing achievements, several challenges arise, one of them is the ability of the autonomous car to perceive its environment in order to properly evaluate the situation with regards to the road environment. Part of this situation evaluation is the knowledge about ego-localization. In the broadest sense, ego-localization is a meaningful concept that can be related to different problematics. However, one interpretation of ego-localization consists of the knowledge of three key components: the road on which the vehicle is traveling (Road Level Localization (RLL)), the ego-lane position (Ego-Lane Level Localization (ELL)), and the lane on which the vehicle is traveling (LLL). Therefore, a reliable ego-localization system has to fulfill the localization's requirement of each of these components.
The objective of this Ph.D. work is to propose a unified, generalized and modular localization system architecture that tackles every aspect of the localization system. In addition that, a focus is given on opensource map OpenSteetMap (OSM) to demonstrate that even a low-cost map can be used to obtain an accurate localization. To do so, an end-to-end framework composed of several interconnected components is presented. This framework is responsible of providing a localization solution on a digital map by developing a robust map-matching algorithm. Furthermore, it permits the localization of the ego-vehicle with respect to ego-lane by proposing a top-down approach that exploits the priors of the map in order to detect the lane marking. Finally, it determines the lane on which the vehicle is traveling by introducing a modular framework that handles the ambiguities in the lane-level localization. The reliability and the flexibility of the overall proposed architecture and its elementary components have been intensively validated, first, individually using different dataset, and secondly, as a whole solution using a collected dataset in the region of Clermont-Ferrand.

Keywords: Autonomous driving, Localization architecture, Probabilistic framework, Lane detection, Fusion Framework.

RÉSUMÉ

De nos jours, les véhicules autonomes (VAs) sont capables de réaliser des tâches extraordinaires et compliquées. Malgré ces réalisations étonnantes, plusieurs défis se posent, l'un d'eux étant la capacité de la voiture autonome à percevoir correctement son environnement afin d'évaluer correctement la situation dans laquelle elle se trouve. Cette évaluation de la situation repose en partie sur la connaissance de l'égo-localisation. Au sens large, l'ego-localisation est un concept vaste qui peut être lié à différentes problématiques. Cependant, une interprétation de l'égo-localisation consiste en la connaissance de trois éléments clés : la route sur laquelle le véhicule circule (Road Level Localization (RLL)), la position de du véhciule par rapport aux marquages de sa voie (Ego-Lane Level Localization (ELL)), et la voie sur laquelle le véhicule circule (Lane-Level Localization (LLL)). Par conséquent, un système d'ego-localisation fiable doit répondre aux exigences de localisation de chacun de ces composants. L'objectif de ce travail de doctorat est de proposer une architecture de système de localisation unifiée, généralisée et modulaire qui aborde tous les aspects du système de localisation. En outre, l'accent est mis sur l'utilisation de la carte opensource OpenSteetMap (OSM) pour démontrer que même une carte à faible coût peut être utilisée pour obtenir une localisation précise. Pour ce faire, une architecture probabliste dite end-to-end composé de plusieurs composants interconnectés est présentée. Cette architecture est chargée de fournir une solution de localisation sur une carte numérique en développant un algorithme robuste de correspondance de carte (map-matching). De plus elle permet de localiser l'ego-véhicule par rapport à l'ego-voie en proposant une approche descendante qui exploite les informations a priori de la carte afin de détecter les marquages de l'ego-voie. Enfin, elle permet de déterminer la voie sur laquelle le véhicule se déplace en introduisant une architecture modulaire qui gère les ambiguïtés dans le choix de la bonne voie. La fiabilité et la flexibilité de l'architecture globale proposée et de ses composants élémentaires ont été validées , d'abord individuellement à l'aide de différents ensembles de données, puis en tant que solution globale à l'aide d'un ensemble de données collectées dans la region de Clermont-Ferrand.

Mots-clés : Véhicules autonomes, architecture de localisation, architecture probabiliste, détection des voies, architecture de fusion d'information.

CONTENTS

1 General introduction 13
1.1 Background and motivation . . . 13
1.2 History of the autonomous vehicle . . . 14
1.2.1 The dawn of the autonomous car . . . 14
1.2.2 The Darpa Challenge: “the rest is just stamp collecting” . . . 14
1.3 Popular views on autonomous vehicle . . . 17
1.4 Intelligent Vehicles Architecture . . . 18
1.4.1 Perception . . . 19
1.4.1.1 Exteroceptive sensors . . . 19
1.4.1.2 Proprioceptive sensors . . . 20
1.4.2 Localization and mapping . . . 20
1.4.3 Path planing and control . . . 20
1.5 Problem formalization . . . 21
1.5.1 Objective of the thesis . . . 22
1.5.2 Context of this PHD . . . 22
1.6 Thesis scope and contributions . . . 23
1.7 Manuscript outline . . . 24

I State of the art 26

2 Road Level Localization 29
2.1 Introduction . . . 29
2.2 Terminologies . . . 29
2.3 Related work . . . 30
2.3.1 deterministic model approaches . . . 31
2.3.1.1 Geometric algorithms . . . 32
2.3.1.2 Pattern-based algorithms . . . 33
2.3.2 Probabilistic model approaches . . . 33
2.3.2.1 Hidden Markov Model (HMM) . . . 34

2.3.2.2 Conditional Random Field (CRF) . . . 37
2.3.2.3 Weighted graph technique (WGT) . . . 38
2.3.2.4 Particle filter (PF) . . . 39
2.3.2.5 Multiple hypothesis technique (MHT) . . . 40
2.4 Conclusion . . . 41

3 Ego-Lane Level Localization 44
3.1 Introduction . . . 44
3.2 Related work . . . 45
3.2.1 Model-driven approaches . . . 46
3.2.1.1 Pre-processing . . . 47
3.2.2 Feature extraction . . . 49
3.2.3 Fitting procedure . . . 52
3.2.3.1 Tracking procedure . . . 53
3.2.4 Learning approaches . . . 54
3.3 Conclusion . . . 58

4 Lane Level localization 61
4.1 Introduction . . . 61
4.2 Related Work . . . 62
4.2.1 Map aided approaches . . . 62
4.2.2 Landmarks approaches . . . 65
4.3 Conclusion . . . 67

5 Data-sets for autonomous vehicle 69
5.1 Introduction . . . 69
5.2 Survey on datasets . . . 70
5.3 How to evaluate a localization algorithm? . . . 73
5.4 Conclusion . . . 76

II End-to-End Probabilistic Ego-Vehicle Localization Framework 80

6 Probabilistic Map Matching Algorithm (PMMA) 83
6.1 Problem Formulation . . . 83
6.2 OpenStreetMap Dataset . . . 84
6.2.1 Nodes . . . 85

6.2.2 Ways . . . 85
6.2.3 Relations . . . 85
6.3 Proposed solution . . . 86
6.3.1 Pre-Processing . . . 86
6.3.2 Discrimination . . . 88
6.3.2.1 Distance to road . . . 88
6.3.2.2 Angle difference . . . 89
6.3.2.3 Speed difference . . . 89
6.3.3 Selection Stage . . . 90
6.3.3.1 Probabilistic criterion based on Euclidean distance . . . 90
6.3.3.2 Probabilistic criterion based on Mahalanobis distance . . . 91
6.3.3.3 Probabilistic criterion based on the probability of belonging to a segment . . . 92
6.3.4 Functioning of the algorithm . . . 93
6.3.5 The enhanced Probabilistic Map Matching Algorithm (PMMA) . . . 95
6.4 Results and discussion . . . 98
6.5 Conclusion . . . 102

7 Ego-lane Level localization solution 103
7.1 Problem formulation . . . 103
7.2 Proposed solution . . . 104
7.3 Recursive Information-Driven Algorithm (RIDA) . . . 105
7.3.1 Initialization stage . . . 105
7.3.2 Selection of the best Region Of Interest (ROI) . . . 109
7.3.3 Detection . . . 112
7.3.4 End . . . 112
7.3.5 Tracking step . . . 113
7.3.6 Canonical example . . . 114
7.4 Results and discussion . . . 117
7.4.1 The datasets . . . 117
7.4.2 Construction of the dataset . . . 117
7.4.3 Quantitative results . . . 118
7.4.4 Qualitative results . . . 119
7.5 Conclusion . . . 120

8 Lane Level localization solution 122

8.1 Problem formulation . . . 122
8.2 Modular Probabilistic Framework For lane level localization . . . 123
8.2.1 Adjacent lanes extrapolation . . . 123
8.2.2 Bayesian network for ego-lane determination . . . 124
8.2.2.1 Bayesian Network using only adjacent lanes detection (bn+ALD) . . . 125
8.2.2.2 Bayesian Network with the addition of the adjacent vehicle detector (bn+ALD+AVD) . . . 126
8.2.3 Hidden Markov Model Lane-Level Localization . . . 127
8.2.4 Transition probability (Lane change probability) . . . 128
8.3 Real-world experimental results . . . 130
8.3.1 Quantitative results . . . 130
8.3.2 Qualitative results . . . 131
8.4 Conclusion . . . 131

9 End-to-end coherence results 134
9.1 Used acquisition platform . . . 135
9.2 Overview of the dataset . . . 135
9.2.1 Structure of the dataset . . . 136
9.3 Performed experiments . . . 136
9.4 Results and discussion . . . 137
9.4.1 Functioning of the end-to-end algorithm . . . 137
9.4.2 Results . . . 140
9.5 Conclusion . . . 144

General conclusion and perspectives 145

10 Annex 150
10.1 Hidden Markov Model(HMM) . . . 150
10.2 Bayesian Network (BN) . . . 152
10.2.1 Probability properties . . . 152
10.2.1.1 Bayes' Rule . . . 153
10.2.2 Bayesian Networks . . . 153
10.3 Product of the Gaussian . . . 154

Bibliography 159

List Of Figures

1.1 The odyssey of autonomous vehicle with some key historical moments illustrated for both academic and industrial world. 15

1.2 Levels of driving automation by SAE International's new standard J3016 (SAE, 2014) 16

1.3 A traffic jam situation in a famous roundabout in Paris (place de l'étoile) . . 18

1.4 Left images are correctly predicted sample, the predicted images are a distortion (in center). All images in the right column are predicted to be an "ostrich" [81] 18

1.5 The basic navigation loop for AVs 19

2.1 Map-matching diagram with some applications [90] 30

2.2 Map-Matching (MM) classification based on our review to the literature. . . 31

2.3 The geometric map-matching algorithms. Grey circles denote a position sample p_i, white circles are road network nodes and black dots are the results of the matching using each method [90] 32

2.4 Architecture of the DeepMM [154] that takes as input raw GNSSdata in order to choose the correct road. 34

2.5 HMM based on Hummel [30]. (a) is a road network situation, (b) is the corresponding HMM, the arrows between the nodes represent the connectivity of the network. 35

2.6 Example of an ambiguous map matching situation resolved with the cubic interpolation (right image). The red arrows represent the vehicle's heading estimation. This example was presented originally in [48]. 36

2.7 An example of the "route distance". Three road segments, r_1, r_2, and $r3$, and two measured points, z_t and z_{t+1}. The first measured point, z_t, has candidate road matches at x_{t1} and x_{t3} . Each match candidate results in a route to $x_{t+1,2}$, which is a match candidate for the second measured point, z_{t+1}. These two routes have their own lengths, as does the great circle path between the two measured points. [59] 37

2.8 A CRF for 3 GPS observations. The map on top illustrates the simplified situation of identifying road states and path states given GPS observations in the road network. This requires 5 random variables, $y_1 : \{r_1, r_2\}, y_2 : \{p_1, p_2, p_3\}, y_3 : \{r_3, r_4\}, y_4 : \{p_4, p_5\}, y_5 : \{r_5, r_6\}$, to build the CRF for map matching. Thus, nodes y_1, y_3, y_5 linking with observations (black circles) are point nodes while nodes y_2, y_4 are path nodes [98] 38

2.9 Overview of the st-matching algorithm [58] 39

2.10 Principle hypothesis branching observed on two consecutive measured positions. 40

3.1 Some lane detection systems [11]: (a) Lane departure warning (b) Adaptive cruise control (c) Lane keeping or centering (d) Lane change assist. (image credit [121]). 45

3.2 Model driven approach (top) vs monolithic learning approach (bottom) . . . 46

3.3 Use of absolute camera calibration to project real-world quantities, such as the position of the sun on the image (credit [54]). 47

3.4 The pre-process fro eliminating non-relevant part of the image. (credit [51]). 48

3.5 ROI definition using the probabilistic model. (Image credit [18]). 49

3.6 Proposed algorithm to estimate road priors using a digital map. (a) Acquisition of the geometrical information from the digital map database. (b) The modeling of the road depending on the acquired information. (c) Road shape projection in the BEV (credit [83] 50

3.7 ROI estimation (in green) based on the vehicle speed.(Image credit [100]). 50

3.8 Comparison of different features extractors. The first column is the original image. The second column is for ground truth. Third-eighth columns (from left to right): global threshold, + - gradients, steerable filters, top-hat filter, local threshold and symmetrical local threshold (image credit [49]) 51

3.9 Result of a BEV image (credit [142]). 51

3.10 Feature extraction through a modified version of Otsu method. Blue lines are the road bounds estimated by curb detector and yellow points correspond to the detected road marking. (Image credit [87]). 52

3.11 Two lane marking models on the same rural scenario. Images (a) present a 3D B-spline model and images (b) illustrates a planar clothoid model(image credit [65]). 53

3.12 The main pipeline of the proposed Algorithm of Golan et al [71].(credit [71]). 55

3.13 The general pipeline of a CNN.(credit [159]). 56

3.14 The general pipeline of the DVCNN(Image credit [102]). 56

3.15 The setup of the laterally-mounted camera looking downward onto the lane marking(credit [102]). 57

4.1 Types of maps generally used in intelligent transport system: (a) HERE map (© HERE), (b) OpenStreetMap (© (OpenStreetMap), (c) mesoscale map used in [115]. Details in text. 63

4.2 Framework proposed by Kang *et al.* [163] 64

4.3 Framework proposed by lee *et al.*, [173] 65

4.4 Framework proposed by Nieto *et al.*, [47] 66

4.5 The semi-fixed Bayesian Network proposed by Popescu *et al.*, [74] 66

4.6 The LLL hypothesis of the ego-vehicle (in blue) and the adjacent vehicles detected (in red) (image credit [106]). 67

5.1 The recording vehicle used for the KITTI dataset (left). The mounting position of all the sensors (red) with the respect to the vehicle frame. Heights are noted in green and transformation in blue (image credit [70]). 70

5.2 (a) different scenarios for the Culane dataset, (b) proportion of the different scenarios (image credit [138] 71

5.3 (a) represents the labeled lane image produced by the automated labeling framework. (b-d) shows three representation for the lane marking. (image credit [141] 72

5.4 (a) The two driving test circuits (b) The vehicle used for the experiment (Image credit [158] 72

5.5 The recording vehicle used for the AppoloScape dataset (left). In the right, an example of color image (top), 2D semantic label (middle), and depth map for the static background (bottom) (image credit [129]). 73

5.6 A graphical illustration to help understand the previous example.(image credit [143] 75

5.7 End-To-End Probabilistic Framework For Ego-Localization proposed in this work. Each part of the framework are the focus and the main contributions of the Ph.D. thesis. 82

6.1 Example of an OSM Way with its corresponding tags, keys and values. . . . 86

6.2 Flowchart of the proposed PMMA [164]. The inputs are highlighted in yellow, the outputs in green, and the processing stages are in blue. 87

6.3 OSM map before preprocessing stage Figure 6.3a and after preprocessing stage (elimination of the ways that does not represent roads on which the ego-vehicle can travel) Figure 6.3b 87

6.4 Closest distance between a geographical object p and a segment $s = (n_1, n_2)$. 89

6.5 Difference of angles α_1, α_2 between GPS trace and Ways W_1, W_2 89

6.6 Example of calculation of the Mahalanobis distance. $\mathbf{x} = (x, y, \theta)^T$ the current state vector, and $\mathbf{x}_s = (x_s, y_s, \theta_s)^\top$ is the vector representing the orthogonal projection on the segment. 92

6.7 Illustration of the probability of the segment (in red) to belong to the expected area of presence of the ego-vehicle, the blue area corresponds to the probability we are looking to compute. 93

6.8 Initialization step for the MM algorithm. Red lines represent the Way extracted from OSM. 94

6.9 Preprocessing step with the elimination of roads that are not suitable for MM. 94

6.10 Discrimination stage depending on the distance (a) and the orientation of the vehicle (b). 95

6.11 Selection of the right Way in blue. 95

6.12 Modeling of the Hidden Markov model Map-Matching (HMM-MM) algorithm, with Sc_t the state space at time t, s_t the segment candidate at time t and O_t the observations at time t. Note that the number of candidate segments at each time may vary. 96

6.13 Example of road network to illustrate the transition probability calculation. In this example the number of Way candidates is the same between two consecutive frames 97

6.14 Illustration of calculation for transition probability λ^s, the vehicle is illustrated by the box with black triangle, ψ represents the segment heading, θ the vehicle's heading, and $\mathbf{P}'$ is the projection of the vehicle position on the segment $s = [n_1, n_2]$. 98

6.15 The route traveled in the region of Paris illustrated with blue color in OSM. Around 100 km have been traveled for a total of 6596 GPS frames. 99

6.16 Lanes estimation compared to ground truth for every criterion used in the map matching [133]. 100

6.17 Scenarios where the map matching algorithm selects the wrong segment and leads to a false lanes number estimation 101

7.1 All different steps of the Recursive Information-Driven Algorithm (RIDA) for ego-lane marking detection. 105

7.2 The robot navigates on the lane of width L_w modeled as a polynomial equation $y(x)$. The left and right borders of the lane are respectively approximated with the polynomials $y_l(x)$ and $y_r(x)$. 106

7.3 Overall algorithm for the proposed ego-lane detection. (a) The lidar provides pointclouds. (b) Using an IMU, several pointclouds are accumulated. 108

7.4 Intensity image of the lidar points with the model initialization of the presented ego-lane marking model. The green line is used for the mean value of the probabilistic model represented by the vectors $\mathbf{y}_{l,r}$, and the blue rectangles are used to represent the uncertainty $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}$. 109

7.5 Projection of the initial probabilistic model $(\mathbf{u}', \mathbf{\Sigma}_{\mathbf{u}'})$ on the image. The mean values of the lane marking $\mathbf{u}'$ are presented in red, blue illustrates the dispersion around these values (1 σ). Finally, green boxes are the ROI for each lane marking, in this example the number of ROI is 9 for each lane marking. 110

7.6 Bayesian Network used for the confidence estimation. The yellow nodes are the input nodes, the purple nodes are the nodes we are seeking to estimate. 111

7.6 All steps for the RIDA. In the upper part of each image, red is used for the confidence about the probabilistic model, blue for the entropic gain, and the green line represents the threshold for which the recognition is considered finished. In the lower parts of each image the lane marking are illustrated in blue, the uncertainties about these marking is show in red, green is used for theROI and yellow for the selected ROI. Details about each image are given in text. 116

7.7 Process for the evaluation of the detection lane, first a BEV (Bird's Eye View) image is produced with its associated ground truth in blue (a), the result of the detection is transposed in the BEV image in green (b). 118

7.8 RMSE of the sequence 27 and of the sequence 28. The color code used is the follow: Orange stands for sequence 27 and Blue for sequence 28 . . 119

7.9 Some examples of detections using the lidar, the results were projected into the image for a visual comparison with a camera solution. (a) shows a case where the lidar and camera solution correctly detects the ego lane, (b) shows a case where the camera fails due to high brightness, (c) the lidar fails where the camera does not, in (d) both of the solution fail. The color code used is as follows: green for the mean value of the ego-lane marking using lidar and blue for camera. An additional video is provided in shorturl.at/bktvE 120

8.1 Perspective view, ego-lane in solid lines and hypothesized edges in dotted lines (case three lanes road) 124

8.2 Confidence intervals for adjacent hypothesized edges for a two lanes road. Green represents the estimated edges of the ego-lane and blue illustrates the confidence intervals for the possible positions of the adjacent lanes marking. 124

8.3 Graphic representation of the general architecture of the Bayesian Network used for the ego-lane determination 125

8.4 Results of ego-lane determination using the BN with only the adjacent lanes' detection. In (a) the BN success as the marking are correctly detected, in (b) and (c) the adjacent lanes are not detected, which leads to a false result. In all figures, green represents ego-lane marking and yellow represents adjacent lanes marking detected 126

8.5 BN with adjacent lanes and vehicles detection (case 4 lanes road) 126

8.6 Results of ego-lane determination using the BN with the adjacent lanes and vehicle detection. For the case (a), the addition of the vehicle detector does not change the result, since no vehicle is detected. However, in (b) and (c) the vehicle detector leads to a correct determination of the ego-lane. In all figures, cyan indicates the bounding box of vehicles detected on the image. 127

8.7 Hidden Markov Model for n lanes case, with $L = \{l_1, l_2, ..., l_n\}$ being the set of hidden states and $O = \{e_1, e_2, ..., e_n\}$ being the set of observations resulting from the BN. 128

8.8 Estimation of the lateral shift μ_{y_0} and its variance σ_{y_0} (3 σ) in case of lane change. 129

8.9 Lane change probability, with the blue area representing the right change probability. 129

8.10 Examples of correct and incorrect ego-lane determination on the 4-lanes datasets.The first row only correct classification are given. On the other hand, the second row shows incorrect classification. In all cases, the ego-lane marking is shown in green and the adjacent vehicle are shown in Blue, finally, the yellow is used for the adjacent lane marking detected. An additional video is provided in shorturl.at/fjD06 133

9.1 The vehicle used VELAC in Figure 9.1a, and the mounted sensor for the experiment in Figure 9.1b 135

9.2 Path traveled in blue for a total of 5 km (image taken from Google) 136

9.3 Topics stored in the bag using rqt_bag. Relevant topics are highlighted in red. 137

9.4 First step of the RLL with the display of the ways from OSM database. The drawing have been centered around the estimated vehicle position. 138

9.5 Selection of the Way on which the vehicle travels using the PMMA algorithm presented in Chapter 6. 138

9.6 Projection of the polynomial model into the image space. The means values are represented in red, the interval of research in blue and the ROI in green. 139

9.7 Detection of the ego-lane marking in blue, and estimation of the road's parameters. 139

9.8 Area of research for the adjacent lane marking in red in which the detection will be performed. 140

9.9 LLL performed taking into account the adjacent lane detection results in cyan, and the detector failure rate. The leftmost lane is noted $Lane_1$, while the rightmost lane is noted $Lane_3$ 141

9.10 Example output of the proposed end-to-end algorithm. The RLL output is presented in the upper left of the image, the lateral shift of the vehicle in the lower left of the image, and the LLL in the right side of the image 142

9.11 A sequence of outputs of the proposed end-to-end algorithm in a lane change scenario. In Figure 9.11a the vehicle is traveling in the third lane (the rightmost lane). In Figure 9.11b the vehicle starts the left lane change, as a consequence the estimated lateral position go to $-L_w/2$. In Figure 9.11c and Figure 9.11d the vehicles completes the lane change and is now traveling the lane number 2. 143

9.12 Computation time in ms for each part of the presented algorithm: RLL in Light-blue, ELL in pink and LLL in light-green, with mean values of $244.53ms$ for the RLL, $58.74ms$ for the ELL and $143.68ms$ for the LLL. Since the computation time for the ELL and LLL depends on the observability conditions of the markings lanes in the image, the dispersion in time is much bigger than the RLL. 144

9.13 Concept of the addition of DL with model-based technique in order to have a better road scene representation and hence a better ego-localization. . . 148

9.14 The new experimental vehicle acquired by Insitut Pascal 149

10.1 Example of the umbrella example, the transition model is $P(Rain_t|Rain_{t1})$ and the sensor model is $P(Umbrealla_t|Rain_t)$ (image credit [66] 151

10.2 Example of a Bayesian network, showing both the topology and the conditional probability tables (CPTs). In the CPTs, the letters B, E, A, J, and M stand for Burglary, Earthquake, Alarm, JohnCalls, and MaryCalls, respectively (image credit [66] 154

10.3 We want to know the probability that the segment [AB] belongs to the Gaussian zone 155

List Of Tables

2.1 Perfomance of four map-matching methods [17] 32

2.2 Summary of MM algorithms in terms of uncertainty-proof, matching break, integrity indicator and run time. 42

3.1 Various lane fitting models presented in the litterature. 54

3.2 Summary of DL algorithms benchmarked in Tusimple in terms of accuracy and F1 score 59

3.3 Summary of DL algorithms benchmarked in Tusimple in terms of F1 score, the results showed stands for the total of all classes of the Culane 60

5.1 Data-sets for localization system. Only the relevant datasets related to our work are presented. The abbreviation used are the following: Vi: Video, Li: LiDAR, U: Urban, H: Highway, Px: Pixel 74

5.2 Requirements needed to evaluate each part of the localization system (RLL, ELL, and LLL), with the corresponding datasets available with respect to the Table 5.1. The abbreviation used are the following: U: Urban, H: Highway, R: Rural 77

6.1 Used OSM tags 86

6.2 Transition matrix for example shown on Figure 6.13 97

6.3 Results for the accuracy rate on lanes number estimation for every criterion (February 2018) 100

6.4 Results for the accuracy rate on lanes number estimation for every criterion (April 2018) 101

7.1 Parameters used for our experiments 118

7.2 Results of RMSE for our experiments on the KITTI dataset. 119

8.1 Classification accuracy for ego-lane determination. bn + ALD refers to the BN fed with the adjacent lanes' detection, bn+ALD+HMM indicates the HMM with the corresponding BN, bn+ALD+AVD refers to the BN feed with adjacent lanes and vehicles detection and bn+ALD+AVD+HMM refers to the HMM with the corresponding BN. 131

9.1 Map Matching results on the entire datasets without and with the HMM . . . 142

9.2 Classification accuracy for LLL. BN+ALD refers to the BN feed with the adjacent lanes detection, BN+ALD+HMM indicates the HMM with the corresponding BN, BN+ALD+AVD refers to the BN feed with adjacent lanes and vehicles detection and BN+ALD+AVD+HMM refers to the HMM with the BN 142

List Of Abbreviations

- **ADAS:** Advanced Driver-Assistance Systems.
- **AVs:** Autonomous Vehicles.
- **BEV:** Bird' Eye View.
- **BN:** Bayesian Network.
- **CIFRE:** Convention industrielle de formation par la recherche.
- **CNN:** Convolutional Neural Network.
- **CRF:** Conditional Random Field.
- **DARPA:** The Defense Advanced Research Projects Agency.
- **DL:** Deep Learning.
- **EKF:** Extended Kalman Filter.
- **ELL:** Ego-Lane Level Localization.
- **FEV:** Field Of View.
- **FN:** False Negative.
- **FP:** False Positive.
- **GNSS:** Global Navigation Satellite Systems Global.
- **GPS:** Global Positioning System.
- **HMM:** Hidden Markov Model.
- **IMU:** Inertial Measurement Unit.
- **LLL:** Lane-Level Localization.
- **MHT:** Multiple Hypothesis Technique.
- **MM:** Map-Matching.
- **OSM:** OpenSteetMap.
- **PF:** Particle filter.

- **PMMA:** Probabilistic Map Matching Algorithm.
- **PMMA:** Probabilistic Map Matching Algorithm.
- **RIDA:** Recursive Information-Driven Algorithm.
- **RLL:** Road Level Localization.
- **RNN:** Recurrent Neural Networks.
- **ROI:** Region Of Interests.
- **SLAM:** Simultaneous Localization And Mapping.
- **TN:** True Negative.
- **TP:** True Positive.
- **WGT:** Weighted Graph Technique.

LIST OF MY PUBLICATIONS

[1] KASMI, Abderrahim, DENIS, Dieumet, AUFRÈRE, Romuald, et al. Map matching and lanes number estimation with openstreetmap. In : 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018. p. 2659-2664.

[2] KASMI, Abderrahim, DENIS, Dieumet, AUFRÈRE, Romuald, et al. Probabilistic framework for ego-lane determination. In : 2019 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2019. p. 1746-1752.

[3] KASMI, Abderrahim, LACONTE, Johann, AUFRÈRE, Romuald, et al. An Information Driven Approach For Ego-Lane Detection Using Lidar And OpenStreetMap. In : 2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV). IEEE, 2020. p. 522-528 **(Selected for the best Student Paper Award)**

[4] KASMI, Abderrahim, LACONTE, Johann, AUFRÈRE, Romuald, et al. End-to-end probabilistic ego-vehicle localization framework. in IEEE Transactions on Intelligent Vehicles, vol. 6, no. 1, pp. 146-158, March 2021, doi: 10.1109/TIV.2020.3017256.

1

GENERAL INTRODUCTION

Contents

1.1 Background and motivation 13
1.2 History of the autonomous vehicle 14
1.2.1 The dawn of the autonomous car 14
1.2.2 The Darpa Challenge: “the rest is just stamp collecting” 14
1.3 Popular views on autonomous vehicle 17
1.4 Intelligent Vehicles Architecture 18
1.4.1 Perception 19
1.4.2 Localization and mapping 20
1.4.3 Path planing and control 20
1.5 Problem formalization 21
1.5.1 Objective of the thesis 22
1.5.2 Context of this PHD 22
1.6 Thesis scope and contributions 23
1.7 Manuscript outline 24

1.1/ BACKGROUND AND MOTIVATION

We live in an age of extraordinary computer machines. Nowadays, computer machines can simulate complex physical phenomena, beat expert players in games like AlphaGo [111], and even perform a dance choreography [1]. Notwithstanding these amazing achievements, many tasks that seem easy for humans remain difficult to perform by robots.

The task of driving a car, for example, not only makes the most sophisticated machines and computer models fail but also challenges the worldwide research and industrial community working on Autonomous Vehicles (AVs). What is apparent is that, if AVs are going to replace traditional cars in every day’s life, they will have to deal with all the uncertainties and complexities of our dynamic, and often chaotic, world.

[1] https://www.youtube.com/watch?v=fn3KWM1kuAw

1.2/ HISTORY OF THE AUTONOMOUS VEHICLE

The recent advances in technology had led the world to excitement about AVs. Most of the current AVs have shown promising results dealing and interacting with the world. However, before getting there, AVs have come a long way. As Theodore Roosevelt said, "the more you know about the past, the better prepared you are for the future". Therefore, let's go down the timeline to when it all started.

1.2.1/ THE DAWN OF THE AUTONOMOUS CAR

The journey of autonomous vehicles (cf. Figure 1.1) has begun since the dawn of automobiles themselves. It all started in 1925 when Francis Houdina unveiled the first incarnation of assisted driving. Houdina's driverless car, baptized the American Wonder, first drove "without a pilot" down the streets of New York City [118]. The American Wonder, chaperoned by an operator with a radio control in nearby, navigated in the streets of New York City but, ultimately, collided with another vehicle. Ironically, the latter was containing photographers that came to cover the event. The enthusiasm for autonomous vehicles continued after the second world war. In the 1950s, General Motors implemented an automatic system for speed control and variable spacing [4]. Cars with such systems were capable to navigate on highways with the use of receivers that were able to detect special tracks installed in the road.
According to the scientific literature, the first automated vehicle was built in Japan in 1977, within the framework of the CACS (Comprehensive Automobile Traffic Control System) project. Under the supervision of Professor S. Tsugawa [6, 10], demonstrations were carried out with a vehicle capable of navigating in a lane on its own while using a camera that detects lane markings. The vehicle was successfully driven under various road environments at a speed within 30 Km/h. Nevertheless, it was until 1988, with a European research initiative that the first autonomous car was developed by Ernst Dickmanns [8]. This initiative paved the way for new research projects, such as PROMETHEUS (PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety (1987-1995)), which aimed to develop a fully functional autonomous car. In 1994, the VaMP driverless car [13] resulting from the PROMETHEUS work managed to drive 1,600km, out of which 95% were driven autonomously.
Notwithstanding the latest success made toward a fully self-driving car, the latter one remains problematic and many challenges were still unsolved. Meanwhile, Advanced Driver-Assistance Systems (ADAS) reached commercial success. Mitsubishi presented the first distance control system based on LiDAR [14]. Following this momentum, in 1999 Mercedes-Benz implemented the first radar-assisted adaptive cruise control. In the early 2000s, most of the vehicles were equipped with a navigation system (Global Positioning System (GPS)).

1.2.2/ THE DARPA CHALLENGE: "THE REST IS JUST STAMP COLLECTING"

The Defense Advanced Research Projects Agency (DARPA) Grand Challenges gave a new impulse to the research in autonomous vehicles and on the design of complex system architecture to autonomous driving. The major challenge in comparison to previous demonstrations is that there was no human intervention during all the races. Whereas

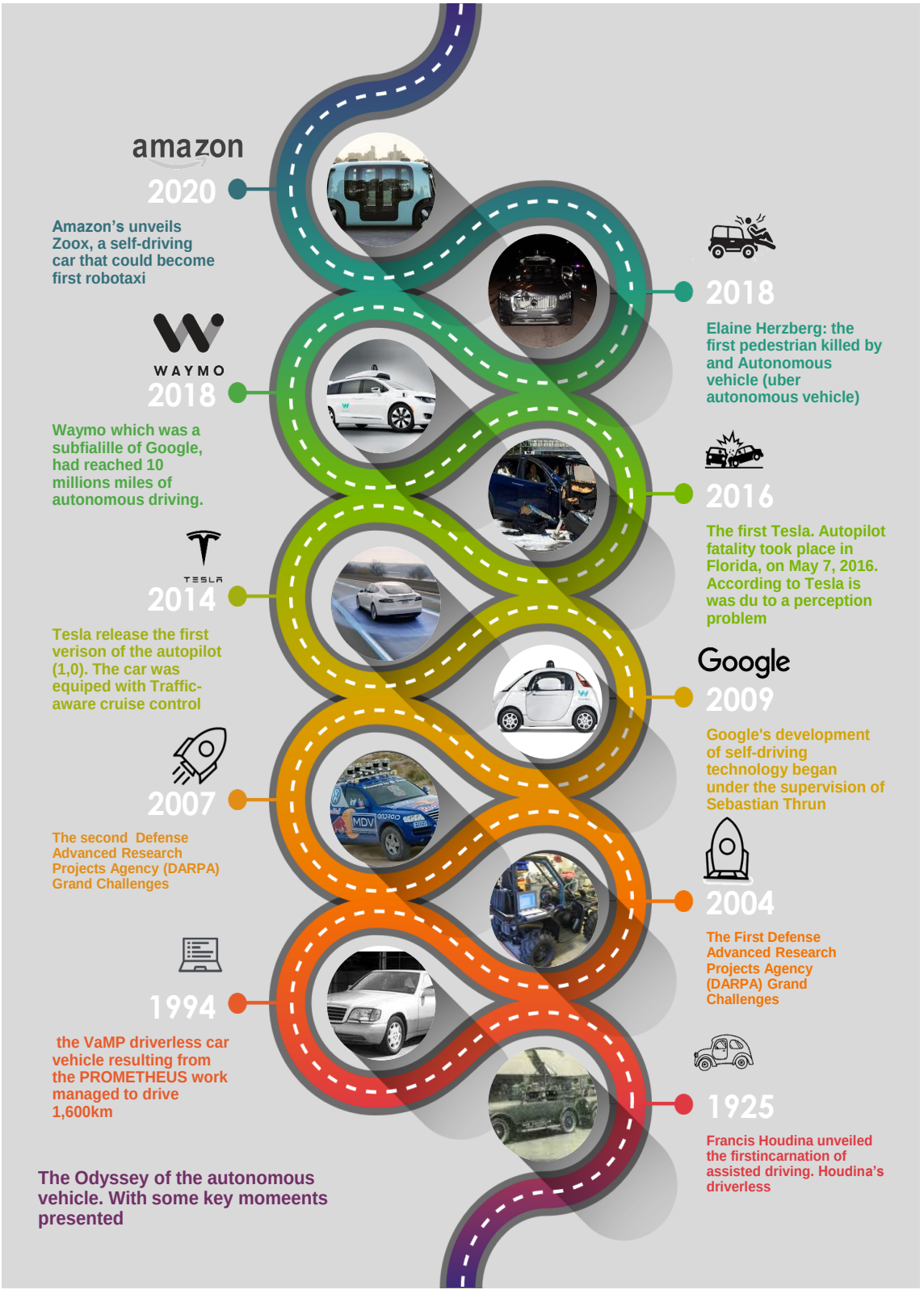


Figure 1.1: The odyssey of autonomous vehicle with some key historical moments illustrated for both academic and industrial world.

the 2004 and 2005 DARPA Grand Challenges were intended to demonstrate that autonomous vehicles can travel significant distances, the 2007 DARPA Urban Challenge (DUC) was designed to promote and spur innovations in autonomous vehicles in cluttered urban environments. No winning team was declared in the first DARPA edition because none of the robot vehicles completed the race. However, a vehicle named Sandstorm went the farthest and gave a hint on the autonomous vehicle's capabilities to win the challenge [24]. Therefore, a second DARPA Grand Challenge event was scheduled for 2005. In this event, the Stanford's robot "Stanley" finished the course ahead of all other vehicles, and was declared the winner of the DARPA Grand Challenge [36]. In that time, the Stanley team was led by Sebastian Thurn.

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	**No Automation**	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	**Driver Assistance**	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	**Partial Automation**	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	**Conditional Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	**High Automation**	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	**Full Automation**	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

Figure 1.2: Levels of driving automation by SAE International's new standard J3016 (SAE, 2014)

Unlike previous challenges, the 2007 Urban Challenge allowed international participants. Furthermore, the race required to drive through a town while obeying to traffic rules. The CMU team won the race, followed by the Stanford team. A noteworthy event is that the majority of the teams relied on multi-beam LiDAR technology developed by Velodyne. This newly released technology will subsequently give a stimulus to several research works related to occupancy grid map and Simultaneous Localization And Mapping (SLAM) techniques for localization.
Ernest Rutherford said, "All science is either physics or stamp collecting" (we would not want to know what he thinks about computer science, would we?), this quote resumes what happens after the success of the two last editions of the DARPA Challenge (2005, 2007). Indeed, in the common mindset, the majority of the self-driving car challenges were resolved (including perception, localization, decision-making, and path planning) and it was up to the industry to figure out how to commercialize the self-driving car.

However, it was not the classical autonomous manufacturer who took command of the race for the autonomous vehicle, it was the giant tech industry. In 2009, the tech giant Google took the lead in the race to get the first autonomous vehicle on the market. They started their autonomous car program called "the Google car". To achieve this goal, Google hired renowned scientist that participated in the DARPA challenges (Sebastian Thrun).
In 2014, the society of automotive engineers (SAE) released their classification of autonomous driving systems into 6 levels of driving delegation [86] as illustrated in Figure 1.2, it summarizes the standard grading for vehicle automation.
In the same year, Tesla released the first version of its Autopilot 1.0 (which corresponds to level 2 of autonomy). Following the same strategy adopted by Google in 2009, Uber starts its own driving car program in 2015 by hiring several researchers from CMU. Ultimately, both Tesla and Uber witnesses their fatal accidents. In 2016 for Tesla, due to self-driving system dysfunctional which was related to the perception module, and in 2018 for Uber, in which a pedestrian was killed by the Uber autonomous vehicle which was lodged by a driver. In 2018, the google car project became a subsidiary of Google under the name of Waymo. The Waymo car reached over 10 million miles of autonomous driving in that year. Nowadays, several companies have joined the race to the autonomous vehicle, i.e, Nvidia, Daimler, Nio. The latest one is another giant tech Amazon, which has the ambition to release the self-driving car Zoox.

1.3/ POPULAR VIEWS ON AUTONOMOUS VEHICLE

Now, we present the past and the current AVs system. Let's discuss (and sweep) some popular views that may arise from the previous section.

1. The driving task is easy: the underlying belief is that driving is an easy task, that is solvable. Since humans are capable of doing it every day, it has to be easy to formalize this task and thus solve it. Maybe the underlying belief is true, but what if not? A demonstration of a traffic situation in a roundabout that occurs everyday in thousands of cities is shown in Figure 1.3. This kind of situation is hardly formalizable. However, we, as human beings deal with it every day without causing accidents. How do we do it? With glances between drivers and with intuitions on the maneuvers of the others, humans are capable of communicating and interpreting glances, which allow them a fast understanding of the road scene environment. So maybe the driving task is not that easy.

2. The second belief is to say that the task of perception is easy and has been settled. The human brain allows us to perceive and understand a scene in split second. Indeed, nature has had millions of years to make the human brain evolve.
   On the other hand, as pointed out by Christian Szegedy, most promising deep neural networks do achieve high performance on visual and speech recognition problems. However, the output can be difficult to interpret and can have counter-intuitive properties [81]. To illustrate these counter-intuitive properties, he added some distortion to input images that was correctly classified by a neural network (a dog, cf. Figure 1.4), the addition of the distortion falsified the results (the dog was classified as an ostrich). In his study, he went further, by doing the same procedure on a bus image, and the output was again an ostrich. So, No! perception is not an easy task.

Figure 1.3: A traffic jam situation in a famous roundabout in Paris (place de l'étoile)

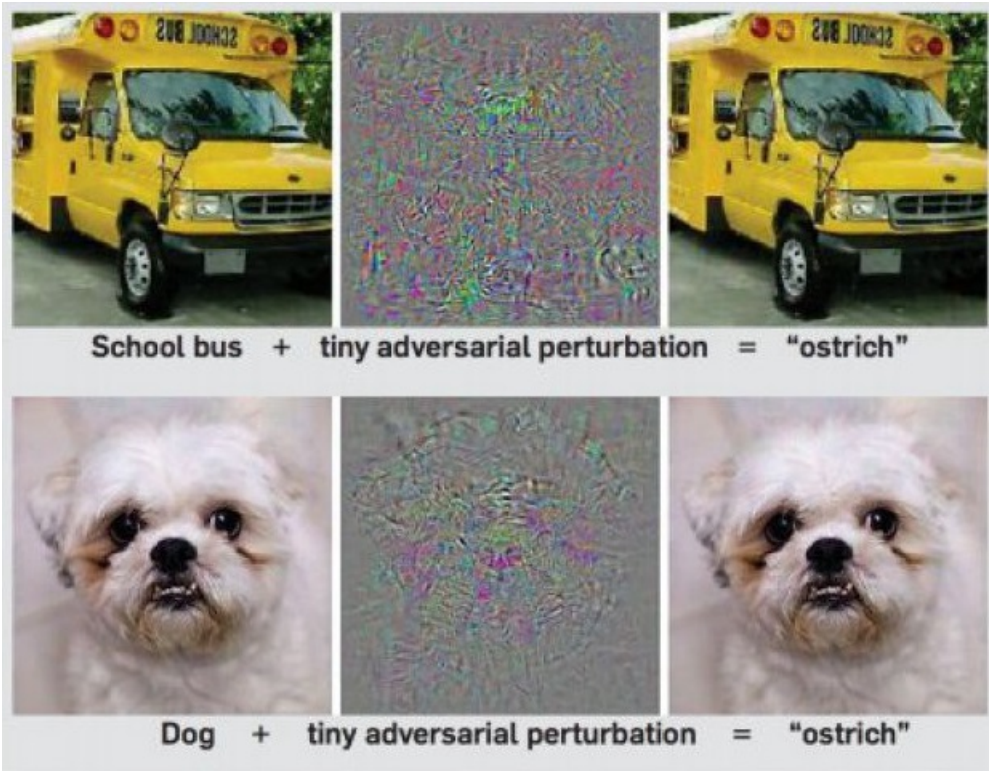


Figure 1.4: Left images are correctly predicted sample, the predicted images are a distortion (in center). All images in the right column are predicted to be an "ostrich" [81]

1.4/ INTELLIGENT VEHICLES ARCHITECTURE

Regardless of the difference between the AVs presented whether, in the industry or in the research, they all must provide a solution to the autonomous navigation problem. In

a global manner, this task is divided into four key elements: perception, localization and mapping, path planning, and control as illustrated in Figure 1.5.

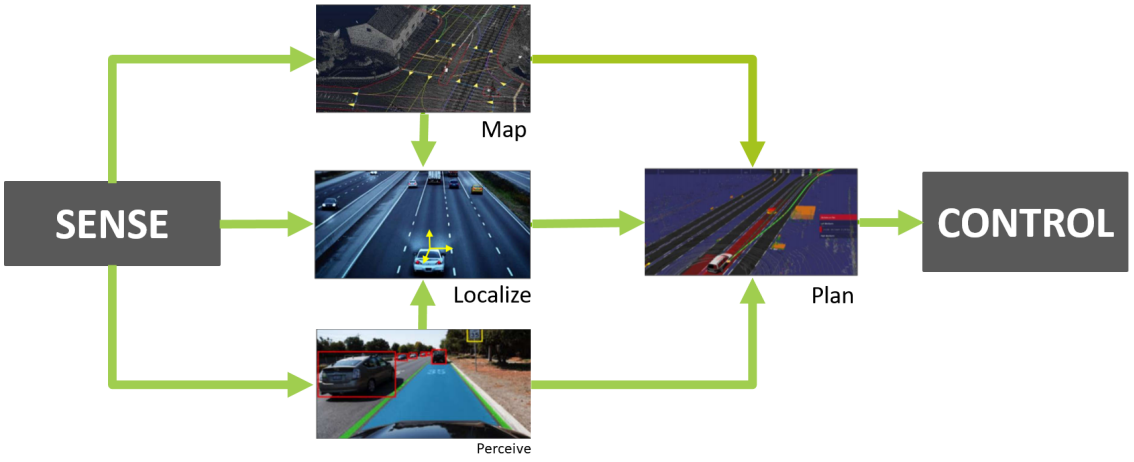


Figure 1.5: The basic navigation loop for AVs

1.4.1/ PERCEPTION

The perception block uses a set of sensors in order to detect relevant features in the environment such as other traffics participants or road signs. To perform such a task, the perception block relies on a perception system that comprises all the sensors embedded in the autonomous car. These sensors can be clustered into two groups: exteroceptive sensors and proprioceptive sensors.

1.4.1.1/ EXTEROCEPTIVE SENSORS

Exteroceptive sensors are used to collect information from the external environment of the vehicle. The most widely used sensors are the following:

1. **Cameras** are the cheapest and most versatile modality for automotive applications and they provide dense information of the environment. For those reasons, they are the most attractive and the most deployed sensors for self-driving cars and ADAS.

2. **Radar** is mainly used to detect obstacles. The technology is based on the measurement of the time of flight between the radar and the object. Since the speed of sound waves is known, the distance is then computed. RADAR sensors have a longer working distance than other sensors due to their longer wavelength but at the cost of reduced accuracy.

3. **Lidar** uses the same technology as radar involving infrared light. The lidar returns dense point clouds representing the environment. Furthermore, the reflectance is also returned by the lidar. Contrary to the radar, the working range of the lidar is smaller. However, the accuracy obtained in terms of richness of information is greater.

The majority of autonomous vehicle systems incorporate data from multiple sensors, allowing them to supplement one while simultaneously solving the limitations of individual sensors, such as the loss of structure information in cameras or the lack of color information in lidar data.

1.4.1.2/ PROPRIOCEPTIVE SENSORS

Proprioceptive sensors provide information about the state of the autonomous car. Indeed, in order to model the autonomous vehicle's trajectory, information about its orientation and speed are necessary. To this end, wheel odometry is used to measure the rotation of a wheel and hereby, is used to estimate the distance covered by the autonomous vehicle.
These proprioceptive sensors are usually low-cost and return uncertain information. Therefore, this uncertainty must be corrected in order to have a reliable estimation of the vehicle's state. Nevertheless, there exists high-accurate proprioceptive sensors; such as high-accurate Inertial Measurement Unit (IMU) that are used for more sensitive applications (e.g. military uses, underwater robotics).

1.4.2/ LOCALIZATION AND MAPPING

The localization and mapping block is responsible to locate the vehicle, first, globally on a map, and then locally on the road.
Global localization is performed with the respect to world coordinates. To this end, Global Navigation Satellite Systems Global (GNSS) are used in order to compute an absolute position in terms of longitude, latitude, and altitude. This localization technology is widely adopted for guidance systems in the automotive industry. Classical Global Positioning System (GPS) receiver have an accuracy order of 5 m. Nonetheless, considering the autonomous car, the lane/road understanding demands in terms of precision [89] vary from one application to another, and therefore, metric precision is not sufficient.
On the other hand, local localization is performed with the respect to the relevant features, that contain contextual information about the road environment, in order to have a local representation of the environment with respect to the autonomous car. These features can be either the outputs of the perception module or they can be stored in digital maps. These digital road maps provide prior information about the road environment without being affected by the limitations of the exteroceptive sensors, and hence, provide information unreachable by these sensors. The range of application of the digital map varies depending on its degree of details (See Section 4.2.1).

1.4.3/ PATH PLANING AND CONTROL

At a later stage, the path planning block exploits the representation given by the previous blocks in order to adopt the safest route for the AVs. The instruction given by this high-level block will depend on the paradigm used for path planning and decision making. But also, on the type of presentation given by the previous components.
At last, the control component delivers the necessary values of control parameters such as acceleration and steering angle for an autonomous car in order to follow the selected

route by the path planning. These two high-level components are out of the scope of this thesis work.

1.5/ PROBLEM FORMALIZATION

Given the global functional architecture of autonomous vehicles. Several challenges arise, one of them is the faultless knowledge of the localization of the autonomous car with regards to the surrounding environment. Depending on the application, the lane/road understanding demands in terms of precision and false alarm rate [89] vary from one application to another. Therefore, a localization system must suits perfectly the localization requirement for each application. The ubiquity use of digital road maps for AVs paves the way for new possibilities and helps to reach more accurate localization positioning. In counterpart, challenges rises on the way to properly use them.
In a global manner, the vehicle's absolute location alone does not offer much information. It must be used in combination with a map representation of the environment to be useful. By and large, there exist two paradigms concerning the mapping/localization process. The first paradigm builds its own road environment representation in a map that stores the obstacles in the road scene, and also the semantic of the scene. For this paradigm, post-processing of raw data from sensors has to be performed in order to create an accurate map. This critical procedure is complex and time-consuming. Thereby, the second paradigm uses a map that is already created. The time process for its creating is then not a problem. However, in this paradigm, the localization system is tributary to the level of details of the map used, and the dynamic obstacles have to be detected using other sensors. Nevertheless, whether one of the two paradigms used, they share the same problematic concerning the uses of the map. Indeed, the localization system needs real-time information from the map. To do so, the localization system has to determine its position regarding the map. To deal with the erroneous data that can be found in both the positioning system and the map, a robust localization algorithm must be developed. This problematic is known as map-matching, and it is a crucial process to use the map properly.
The autonomous car system can not only rely on a map to fulfill its localization requirements. Indeed, it must take into account the dynamic nature of the surrounding environment. Therefore, the system has to exploit the raw data sensors coming from the exteroceptive sensors (i.e., camera, lidar) in order to detect relevant features (i.e., lane marking, other traffic participants). These features gather together will improve the localization accuracy and hence meet the localization requirements needed for higher modules (i.e, decision-making, path planning). With that background, lane markings are the most informative features that can be used in order to have a local localization for the autonomous car. Indeed, these features provide information about the shape of the road but also about the positioning of the vehicle, which is known as lateral positioning. For all these reasons, a robust localization algorithm that is based on a perception module must be developed. The latter has to take into account the uncertainties due to raw data sensors. In addition to that, it has to be able to resolve ambiguities that can occur when detecting lane marking (i.e, the lane position with regard to the road). Finally, the system has to be modular to other sensors' information, or other detectors' information as the number of embedded sensors has been growing strongly in the autonomous industry.

1.5.1/ OBJECTIVE OF THE THESIS

The objective of this Ph.D. work is to propose a unified, generalized and modular localization system architecture, while accounting for the aforementioned characteristics and be able to:

1. **Provide a localization solution on a digital map** by developing a robust map-matching algorithm that takes into account the uncertainties found in both raw data sensors and the digital map. Which implies a mathematical formalization for the uncertainties and ambiguities.
2. **Exploits the digital map** by building a probabilistic model of the road shape that integrates the uncertainty of the map. The model is afterward used a prior for the detection module.
3. **Present a top-down approach for ego-lane localization** that exploits the priors of the map in order to reduce the time complexity. The approach has to be indifferent to the type of sensors used (lidar or camera).
4. **An integrity metric** that ensures the validity of the model in case of unexpected or failure of the perception system, which corresponds to incoherence with the probabilistic model.
5. **A modular framework** that handles the ambiguities in the lane localization. The framework has to be modular for other sensors' detection results.
6. **Experiments result** to validate the effectiveness of each block of the algorithm and the whole architecture. The validation is performed on data acquired from a real vehicle and from available datasets.

In this dissertation, we propose an end-to-end architecture that covers all the localization requirements mentioned. Furthermore, a particular focus will be given to low-cost solutions and openSource resources such as OpenSteetMap (OSM).

1.5.2/ CONTEXT OF THIS PHD

This research work is part of a "Convention industrielle de formation par la recherche (CIFRE)" thesis agreement between Sherpa Engineering and the Institut Pascal. Sherpa Engineering is already working on several themes related to the autonomous car within its R&D department and more particularly on ADAS systems (driving assistance systems). One of the main difficulties in this kind of system lies in the precise localization of the autonomous car in its environment. Basically, localization is an algorithmic process that allows the autonomous vehicle to determine precisely where it is in the world. In order to do so, the exploitation of temporal measurements from different exteroceptive and proprioceptive sensors (Camera, Radar, GPS, IMU, Odometer, Lidar...) allows the development of probabilistic methods (Kalman Filter, Particle Filter, HMM, Bayesian Network...) in order to estimate the state of the object's position with the highest possible accuracy. This task is made more difficult because of the imprecision of the sensor data (Camera, Lidar, GPS...) used in this localization process. Naturally, the literature is teeming with solutions to deal with this problem. However, the cost of these solutions is often exorbitant.

It was therefore natural for Sherpa to be interested in the localization methods developed within the Institut Pascal and more precisely within the PerSyst team, whose work over the last two decades has enabled the implementation of applied solutions for the localization of autonomous vehicles. These solutions are characterized by the use of low-cost sensors and OpenSource resources such as OpenStreetMap. Thus, Sherpa's growing interest in the solutions developed by the PerSyst team resulted in a partnership in the form of a CIFRE thesis entitled "Ego-vehicle localization using multi-sensors and OpenStreetMap data" which started in February 2018.

1.6/ THESIS SCOPE AND CONTRIBUTIONS

The thesis emphasis on localization system for autonomous vehicles. Contrary to other works, a feedback link is made between the localization and the perception module, maximizing the coherence between their outputs. A focus is given on highway road scenarios that contain several lanes. In addition, in this Ph.D. work, we give a focus on an open-source map OpenSteetMap (OSM) to demonstrate that even a low-cost map can be used to obtain an accurate localization.
In this work, we present an end-to-end solution that tackles every aspect of the localization system as presented in the aforementioned section. The end-to-end probabilistic framework is responsible for:

- Localizing the vehicle on a digital map by handling the uncertainties and the ambiguities that occur,
- Exploiting the prior information of the map,
- Localizing the ego-vehicle with respects to the ego-lane marking,
- Monitoring the global coherence of the model,
- Localizing the ego-lane with respect to the road, by introducing a modular probabilistic framework.

Therefore our main contributions are the following:

- **A Probabilistic Map Matching Algorithm (PMMA)** It is proposed the development of a map-matching algorithm that takes as input inaccurate GPS information in addition to uncertain map OSM.
- **A probabilistic representation of road model shape** using prior from the map, which will effectively link the global localization with the local perception of the environment in limiting the focus of interest in the detection procedure.
- **Recursive Information-Driven Algorithm (RIDA) for ego-lane localization** is an adaptation and an upgrade of the existing work based on lane detection at the Institut Pascal [18] which constitutes the baseline of the used Recursive Information-Driven Algorithm. The algorithm is enhanced with the entropy features theory in order to reduce the time complexity and to focus the most informative region in the sensor space. In addition to that, a new Bayesian Network is presented to keep track of the coherence of the global model regarding the detection made in a recursive

fashion. The algorithm outputs an ego-level localization with respect to the ego-lane marking in addition to an estimation of the road's parameters. The RIDA has been used on both camera and lidar sensors data, which proves the genericity of the proposed algorithm.

- **A Modular Probabilistic Framework** designed to handle the localization for the ego-lane in multi-lane ambiguities. It utilizes the output of the RIDA in order to estimate the position of the ego-lane with the respect to the other lanes. To do so, the algorithm first extrapolates the possible zone for lane marking and processes a detection in those regions. Then, a designed Bayesian Network (BN) is used. The graphical representation of the BN eases the determination of the ego-lane position. Afterward, an Hidden Markov Model (HMM) is used to constraint the model and to prevent infeasible outputs by taking into account dynamic constraints of the ego-vehicle. Finally, the proposed solution have been compared on the same dataset to the solution presented in [109].

- **Experimental evaluation of the end-to-end probabilistic framework** The effectiveness of the end-to-end probabilistic framework has been intensively validated on different datasets. First on collected datasets in the region of Clermont-Ferrand, and secondly, on available datasets in the literature.

1.7/ MANUSCRIPT OUTLINE

In addition to the general introduction and final conclusions and annexes, the manuscript is composed of 8 chapters which are decomposed into two parts.

The first part titled ***"State of the art"*** corresponds to the state of the art and bibliographical review on the different localization system architecture and on three building blocks composing these architectures namely: Road Level Localization (RLL), Ego-Lane Level Localization (ELL) and Lane-Level Localization (LLL). The objective of this state-of-the-art part is to lead the reader through the methodologies, concepts, and definitions of several methods for a localization system. This part is composed of the following chapters:

- **Chapter 2**: this chapter details the state of the art on the Road Level Localization (RLL) techniques. Main RLL techniques will be presented and classified.

- **Chapter 3**: in this chapter, we propose to classify the algorithmic techniques independently on the various modalities used (Camera, Lidar, Radar) in order to detect the Ego-Lane Marking (ELL).

- **Chapter 4**: in this chapter a survey on the Lane-Level Localization (LLL) methods is presented

- **Chapter 5**: this chapter details the datasets available for autonomous vehicles and how they should get evaluated.

The second part titled ***"End-to-End Ego-Vehicle Localization using Multi-sensor and a Low Cost Map"*** corresponds to the contributions proposed during this Ph.D. thesis. This part contains four chapters:

- **Chapter 6**: this chapter details the Probabilistic Map Matching Algorithm (PMMA) proposed in this work.
- **Chapter 7**: in this chapter, the Ego-Lane Level Localization (ELL) proposed is introduced. The whole functioning of the architecture is presented on both camera and lidar data.
- **Chapter 8**: in this chapter, the LLL algorithm proposed in this work is detailed.
- **Chapter 9**: this chapter present the results of the whole end-to-end probabilistic ego-vehicle localization framework

Each part begins with a global introduction, in which the details of the parts are explained, and a global conclusion, in which a summary is given for all the previous chapters.

I

STATE OF THE ART

OVERVIEW

A fundamental aspect of a fully autonomous vehicle system is its ability to properly perceive its environment in order to properly evaluate the situation of ego-vehicle and with regards to the road environment. Part of this situation evaluation is the knowledge about ego-localization, which implies the knowledge of some keys localization level components.
In the broadest sense, ego-localization is a meaningful concept that has been widely tackled in the literature in a variety of ways. The current literature is teeming with solutions that address this issue in a variety of manners. However, one interpretation of ego-localization consists of the knowledge of three key components

I **Road Level Localization (RLL):** the road on which the vehicle travels.

II **Ego-Lane Level Localization (ELL)** the position of the vehicle in the lane in terms of lateral and longitudinal position.

III **Lane-Level Localization (LLL)** the position of the host lane within the road (the lane on which the vehicle travels).

For the road level localization, digital maps (Google, OpenSteetMap (OSM), or Waze) are used to perform this task. Global Navigation Satellite Systems Global (GNSS) receivers are used to retrieve the geographic (latitude, longitude, and altitude) coordinates, and a map-matching procedure is performed in order to match the position of the ego-vehicle with the correct road ('link'). However, the accuracy of the localization obtained is in the order of meters. Indeed, according to the Federal Aviation Administration (FAA) GPS Performance Analysis Report [93], the accuracy of a standard GPS device is within $3m$ with a 95% confidence, which can not be sufficient for most ADAS that require a more precise localization.
For some applications like lane-keeping, knowing the road on which the vehicle is traveling is not sufficient. These systems must be informed about the position of the host lane in the road to provide adequate maneuver instruction and maintain vehicle's safety.
Further, autonomous vehicle applications need a more accurate localization, which can be translated by the knowledge of the lateral and longitudinal position of the vehicle in the ego-lane. For instance, overtaking maneuvers need a faultless knowledge of the lateral position of the ego-vehicle with respect to the ego-lane marking in order to decide whether the vehicle should overtake the obstacle or not.
The task of vehicle localization is still challenging for an autonomous vehicle, a complete ego-vehicle localization must perform all the three key components described above. Thus, in this work, an end-to-end solution for ego-localization from Road Level Localization (RLL) to Lane-Level Localization (LLL) is presented. As a consequence, in this part, the state of the art related to these three components is detailed.

2

ROAD LEVEL LOCALIZATION

Contents

2.1 Introduction 29
2.2 Terminologies 29
2.3 Related work 30
2.3.1 deterministic model approaches 31
2.3.2 Probabilistic model approaches 33
2.4 Conclusion 41

2.1/ INTRODUCTION

The ubiquity of positioning devices on the vehicles allows the drivers to know the vehicle's position. However, this position is incorrect due to the inaccurate nature of these positioning devices. To address this problem, a correcting procedure is required. One technique matches the vehicle position with a road network coming from a map. This technique is called Map-Matching (MM). As stated by Quddus *et al.* [39], map-matching not only enables the physical location of the vehicle to be identified but also improves the positioning accuracy if good spatial road network data are available. This means, that Road Level Localization knowledge is determined by MM algorithm. The RLL is also a prerequisite component of several applications (cf Figure 2.1). An exhaustive list of applications was given by Velaga in his thesis [67]. This chapter details the state of the art and bibliographical review on the Road Level Localization (RLL) techniques. RLL techniques will be presented and classified.

2.2/ TERMINOLOGIES

Before presenting the map-matching techniques, we start by formalizing the MM problematic. The MM problematic have been studied over two decades, thus, several formalizations have been proposed. In the following, the definition presented inherits from the one presented in [157].

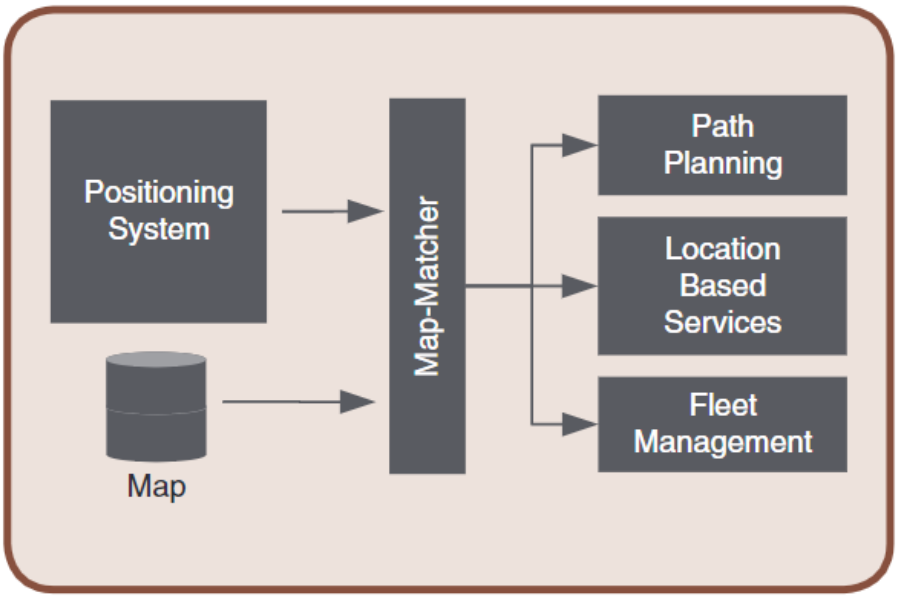


Figure 2.1: Map-matching diagram with some applications [90]

Definition. (Trajectory) A trajectory Tr is a sequence of chronologically ordered spatial points $Tr : p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$ sampled from a continuously moving object. Each point p_i consists of a 2-dimensional coordinate $< x_i, y_i >$ a timestamp t_i, a speed spd_i (optional) and a heading θ_i (optional). i.e.: $p_i =< x_i, y_i, t_i, spd_i, \theta_i >$

Definition. (A map) A map is a directed graph $G = (V, E)$, in which a vertex $v = (x, y) \in V$ represents an intersection or a road end, and an edge $e = (s, e, l)$ is a directed road starting from vertices s to e with a polyline l represented by a sequence of connected segment

Definition. (Map-matching) was defined by Quddus [39]: *"Map-matching algorithms integrate positioning data with spatial road network data (roadway centrelines) to identify the correct link on which a vehicle is traveling and to determine the location of a vehicle on a link"*. Mathematically, MM can be defined as fellows [157]: given a road network $G(V, E)$ and a trajectory Tr, the map-matching finds a route $\mathcal{MR}(Tr)$ that represents the sequence of roads traveled by the trajectory.

2.3/ RELATED WORK

Du to the importance of the RLL, map-matching has been the subject of on-going research since the emergence of Global Positioning System (GPS) in the 1990's [134].
According to the literature, MM techniques can be divided into two categories, namely, online and offline modes. In online mode, the MM procedure is performed in a streaming fashion, which means, that for each point p_i a MM is performed. Consequently, the procedure has to be adequate for real-time application. In contrast, offline MM waits until the trajectory tr is completed in order to perform the MM on the entire trajectory. Hence, this procedure does not care about the real-time requirement. In this work, we will focus only on online MM. Indeed, we want to localize the autonomous vehicle at each sample time. However, it is brought to the attention of the reader that the majority of the techniques

which will be presented can be used for both modes.
Considering online map-matching methods, the most complete and cited survey about the subject was presented by Quddus in [39]. The authors classified the MM techniques from a methodological perspective into four categories, namely geometric, topology, probabilistic and advanced. However, after several years of research on map-matching methods, most of the methods mentioned in the paper have been outperformed and new technologies have emerged. Therefore, this classification is outdated. With a view to bringing up-to-date map-matching techniques, Kubicka and al propose in [134] a survey that classifies the MM methods depending on the application. The study is rich in content and the classification is well organized. However, this study is not very distinctive from the previous one as it follows the previous categorization. Besides, the choice of classification depending on the application does not allow to distinguish between the presented approaches in terms of methodology. Leafing through the literature, it appears that there is still no consensus on how to classify the map-matching methods. However, from our point of view, the classification presented in the survey [157] is the most up-to-date and the most accomplished one. Indeed, it summarizes most of the existing solutions and provides guidance to future research. Therefore, we desire to bring our stone to the edifice and therefore we propose to enhance the proposed classification in [157] with the modest knowledge we gather from previous works.
Hence, we classify the existing MM methods into two different classes: Deterministic Model and Probabilistic Model. Each class is composed of sub-classes. In the following, we will details each category as illustrated in Figure 2.2.

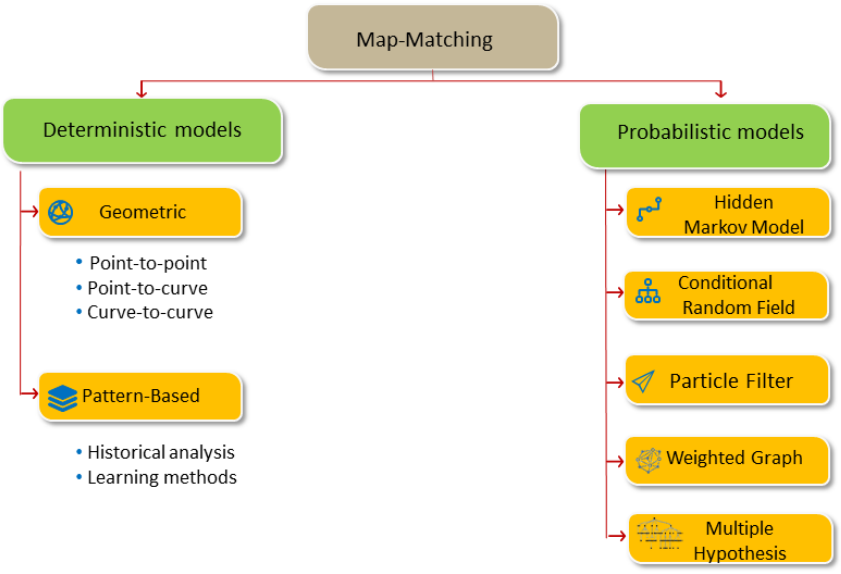


Figure 2.2: Map-Matching (MM) classification based on our review to the literature.

2.3.1/ DETERMINISTIC MODEL APPROACHES

In this approach, the map-matching returns the link that is closest to the trajectory geometrically and/or topologically. In a perfect world, the vehicle's trajectory matches the closest road network topology. Hence, in this model, the main focus is on how to define closeness.

2.3.1.1/ GEOMETRIC ALGORITHMS

The geometric algorithms MM are the most commonly used and oldest methods [17]. These methods were introduced in 1996 by Bernstein and Kornhauser in "An Introduction to Map Matching for Personal Navigation Assistants" [12]. The authors denominate the methods namely, as point-to-point, point-to-curve, or curve-to-curve (cf, Figure 2.3).

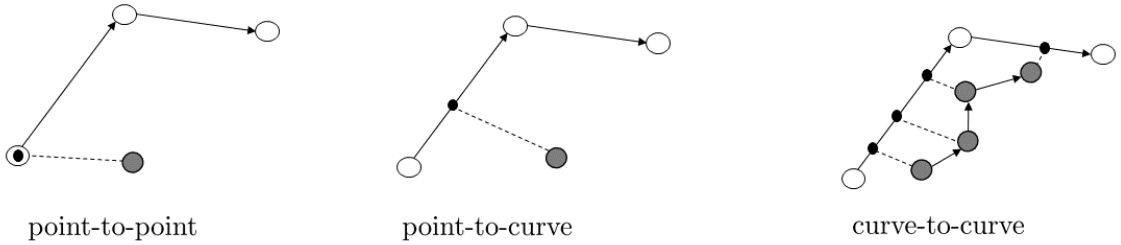


Figure 2.3: The geometric map-matching algorithms. Grey circles denote a position sample p_i, white circles are road network nodes and black dots are the results of the matching using each method [90]

The most elementary approach, so-called point-to-point, matches each position sample p_i to the nearest node e_i. On the other hand, point-to-curves project each position sample p_i to the geometric-closest road. Lastly, curve-to-curve methods match the vehicle' trajectory tr to geometric-closest/similar link in the road-network.
In [17], White *et al.*compared four basic MM methods. The first one is the classical point-to-curve with no consideration of the vehicle state or to the road-network. The second one is a modified version of the point-to-curve. In this modified version the vehicle's heading was taking into account in the matching process. The third method, is an upgrade of the second one, as it takes into consideration the topology of the road network. The last one is the curve-to-curve method. The experiment was conducted on four routes, all the four routes were traveled in the town of Mercer County in New Jersey. Therefore, none of these routes involves highways or arterials. The results are reported in Table 2.1. The matching accuracy presented represents the percentage of the correctly matched samples. For each method, the worst-to-best performance range is noted.

Method	Matching Accuracy
point-to-curve	53 – 67%
point-to-curve, considers heading	66 – 85%
point-to-curve, enforces route contiguity	66 – 85%
curve-to-curve	61 – 72%

Table 2.1: Perfomance of four map-matching methods [17]

All the mentioned methods follow the same paradigm. Indeed, the differentiation appears in the definition of the *closeness*. In the literature, similarity metrics earned a lot of attention. Nonetheless, there is one metric that stands out: the Fréchet distance.
The Fréchet distance was first defined in the thesis of Maurice Fréchet "Sur quelques points du calcul fonctionnel" [1].

An illustration of the Fréchet distance is the following: imagine a person is walking a dog on a leash, the person is walking on a certain curve, and the dog on another one. The assumption is that both have free control over their speeds but are not allowed to go backward. In this particular case, the Fréchet distance of the curves is the minimal length of a leash that is required for both of them to cover the curves from start to finish. Mathematically, the Fréchet distance was defined as follows [1]

$$\delta_F(f,g) = \inf_{\alpha,\beta} \max_{l\in[0,1]} d(f(\alpha(t)), g(\beta(t))) \tag{2.1}$$

With f and g being two parametric curves, and α, β are continuous, monotonic, increasing reparametrizations. This reparametrization allows the monotonicity and continuity of the curves. In other words, in order to compute the Fréchet distance, the reparametrization functions α, β have to be identified.
Alt *et al.* [21] pioneered the Fréchet distance for MM. They were able to find a route whose Fréchet distance to the trajectory is minimal. However, they pointed out one of the major counterparts of the distance. Indeed, if there are outliers to the trajectory, then the solution of the Fréchet distance is not unique, and hence there can be multiple routes that satisfy the optimization. Moreover, the computational demand of the algorithm was too high for a real-time application.
Following this work, Brakatsoulas *et al.* [25] presented a relaxed version of the Fréchet distance in which the increasing property of the parametric curves α, β, namely, the *weak Fréchet distance*. He showed that using the weak Fréchet distance, the algorithm was able to lower the computational requirements of the Fréchet distance. In the same context, further works have been presented with the objective of speeding up the Fréchet distance [37][68] [82]. However, none of these works have been able to resolve the sensitivity to outliers, which is the main drawback of this method.

2.3.1.2/ PATTERN-BASED ALGORITHMS

The pattern-based method is well known in the literature. Indeed, it utilizes the historical resulting of MM to solve new map-matching queries. The assumption is that given a start and an endpoint, people tend to travel on the same trajectory [76]. In that sense, giving a pair of a start and endpoint, and taking into account historical MM results, will lead to finding the most similar trajectories that the vehicle will travel on. Finally, the algorithm will decide on the optimal route base on a scoring function. Instead of using a scoring function algorithm, authors in [135] propose the first deep learning approach that is able to create representations of trajectories. The objective is to capture the route information of each trajectory. In the same manner, Zhao *et al.* [154] presented DeepMM: a deep learning MM system (Cf Figure 2.4). The main drawback noticed in the pattern-based algorithm, even the machine learning one, is the sparsity and disparity of the historical data. Indeed, the disparity of the historical data may not cover all the new query trajectory, which can lead to false MM result.

2.3.2/ PROBABILISTIC MODEL APPROACHES

Although the position data is necessary, it can not be taken as the sole predictor of the vehicle's path. Indeed, naively matching this noisy path to the nearest road, using the

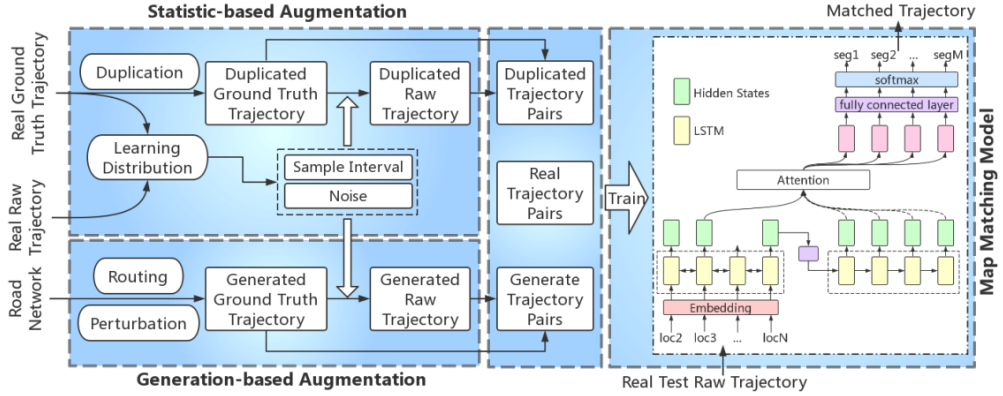


Figure 2.4: Architecture of the DeepMM [154] that takes as input raw GNSSdata in order to choose the correct road.

deterministic metric (cf Section 2.3.1), will eventually result in irrational paths involving non-intuitive and bizarre driving behavior. Hence, a MM algorithm has to consider the reasonableness of a given path in relation to the vehicle dynamics. Therefore, in this section, the MM presented share the same paradigm, which is the probabilistic reasoning, whether it is for the reasonableness of the path, or for the vehicle's dynamic state.

2.3.2.1/ HIDDEN MARKOV MODEL (HMM)

HMM model for MM has been the subject of numerous research studies in connection with tracking problems. The architecture of the MM made it suitable to model the road network topology. The craze for the HMM in the MM problematic was initiated by Hummel [30], resulting in dozens of methods using MM. Most of them were designed for offline map-matching. However, as claimed by Paul Newson and John Krumm in [59], online map-matching is possible using the sliding window technique.
Before listing all the MM techniques used for map-matching, let us understand why the HMM model fits particularly well for the map-matching procedure. To comprehend that, the theory behind the HMM must be presented. Accordingly, a HMM consists of two stochastic processes, the first one is a Markov Chain to model the change of a state vector over time, the state vector is finite. In the literature, the Markov Chain is often graphically visualized with a directed graph. Each node represents a state, and arcs represent changes between states. This change is governed by a probability that describes the transit probability over time, which is called the transition probability. The second process is called the observation space. Indeed, in a HMM, the states of the chain are not visible but observable therefore they are called "hidden". Although the elements of the state vector are hidden, there is a relation between the hidden elements of the state and the observations, this relation is referred to as an emission probability. A more detailed explanation of a HMM is given in annex 10.1.
In the context of map-matching, the state of the system describes the list of the road candidates for each observation. S_t is used to denote the set of routes candidates at time t $S_t = \{l_1, l_2, ..., l_{n_t}\}$ with n_t the number of routes candidate at each time t, S_t is a form of categorical distribution, $S_t \sim Cat(\xi)$.
The state of the vehicle $\mathbf{x_t}$ is the observation. For each candidate route composing the

state space, an emission probability is made $p(\mathbf{x_t}|l)$. A graphic representation of a HMM can be illustrated (cf Figure 2.5 [30]). In this example, the set of states that represent the possible road (or link, hence denoted by l_i) on which the vehicle is located for each estimated vehicle's position (denoted by p_i). The transitions between sets are represented by the arrows. These transitions are governed by the travel possibility between two consecutive sets. This connectivity can be affected by several parameters namely, the road topology, the vehicle's configuration (speed, heading) with regards to the candidate. All these features of the HMM make it naturally fit for the MM process. Now the structure of

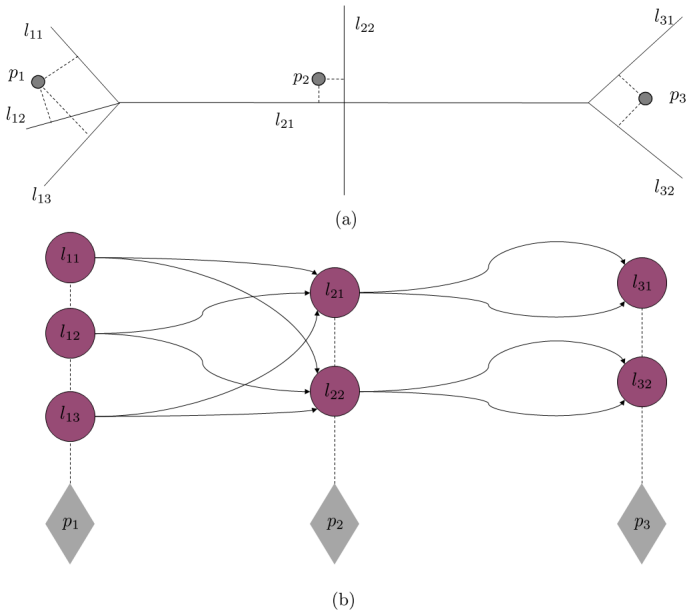


Figure 2.5: HMM based on Hummel [30]. (a) is a road network situation, (b) is the corresponding HMM, the arrows between the nodes represent the connectivity of the network.

the HMM established, the next procedure is to find the most likely road given a trajectory tr. To perform such a task, the standard method is based on the Viterbi algorithm [3]. The algorithm runs on $O(nm^2)$. With n the number of observations (the vehicle's estimated position p_i) and m is the number of states (the possible roads l_i). However, as said earlier, using the sliding window technique can reduce the time complexity. Therefore, the majority of the studies do not differ in the architecture and the representation of the MM task using the HMM. However, they do differ in the definition of the emission probability and the transition probability.
Historically, HMM was first used by Hummel [30] for offline MM. The method presented starts by identifying the most likely road for each sampled vehicle's estimated position p_i. The identification was made using a Mahalanobis distance that minimizes the heading and the distance between the road and the vehicle. Once the set of routes candidates are identified, the backbone of the HMM is the same as the one presented in Figure 2.5. Concerning the definition of the transition probability, is distributed uniformly by taking into account the turn restriction. The method is tributary to the vehicle's heading estimation,

which is estimated by taking into account two consecutive GNSS measurements. In the case of MM with GNSS data, this can be pathological. In particular, when the distance between two GPS measurements is small, the errors on the vehicle's heading are significantly important [134].
The method was later extended by Pink and Hummel [48] resulting in a more robust method. A Kalman Filter was introduced to filter the initial trajectory obtained in order to eliminate outliers. However, the most important contribution was the structuring of the road network using cubic spline interpolation instead of linear interpolation. The motivation behind this modeling is to make the method more robust against the heading errors, which was the main issue with the previous method [30] (cf. Figure 2.6). Besides, it reflects the trajectory shapes that normally vehicles follow.

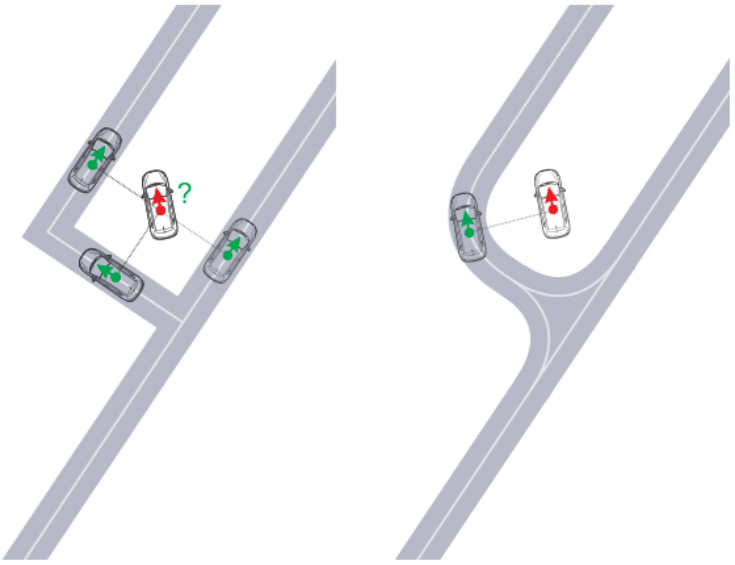


Figure 2.6: Example of an ambiguous map matching situation resolved with the cubic interpolation (right image). The red arrows represent the vehicle's heading estimation. This example was presented originally in [48].

Instead of relying on the vehicle's heading estimation, Newson *et al.* [59] propose a HMM in which the transition probability is governed by a "route distance". The assumption made by the authors is that the correct route has the smallest difference between two consecutive estimated vehicle's positions with the distance along the road-network (cf, Figure 2.7 [59]). The same method was adapted and extended later by Lou *et al.* [117]. Jagadeesh et al [112] enhanced a HMM with the concept of "drivers' route choice". The authors use a route choice model that was modeled using real-world data in order to update the paths generated by a HMM-based online MM.

The HMM based MM achieved an accuracy comparable to the state-of-the-art geometric model [82]. However, HMM based methods suffer from mainly two main drawbacks. The first one is the *the selection bias problem* as pointed out by Hunter et al [78], which is a side effect of the HMM, when giving more weights to the road that are highly disconnected. The second one is that these methods are not robust against missing trajectory samples. In effect, the structure of the transition model in a HMM takes into account the connectivity, physical, and logic between two consecutive sets of route candidates.

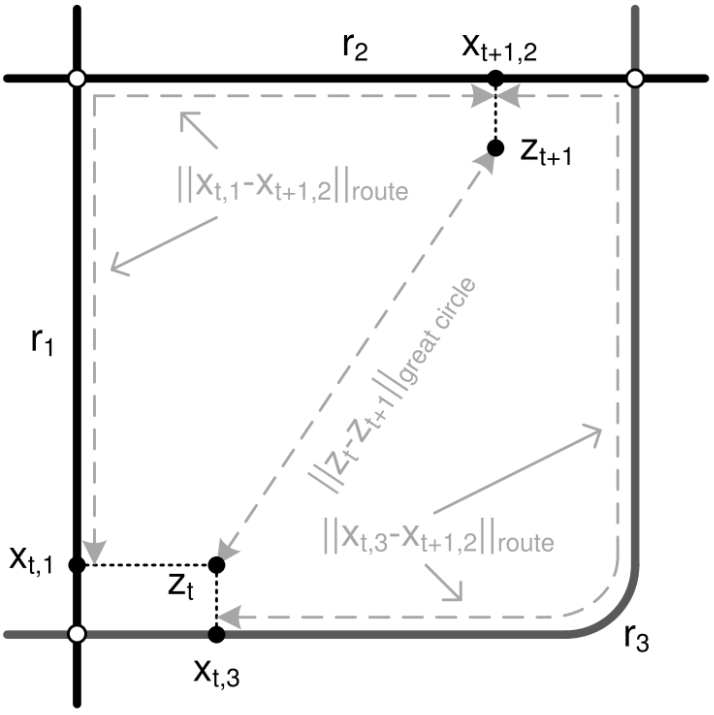


Figure 2.7: An example of the "route distance". Three road segments, r_1, r_2, and $r3$, and two measured points, z_t and z_{t+1}. The first measured point, z_t, has candidate road matches at x_{t1} and x_{t3} . Each match candidate results in a route to $x_{t+1,2}$, which is a match candidate for the second measured point, z_{t+1}. These two routes have their own lengths, as does the great circle path between the two measured points. [59]

However, the discontinuity in position frames will jeopardize the travel possibility between these route candidates. To overcome these issues, more sophisticated methods have been developed.

2.3.2.2/ CONDITIONAL RANDOM FIELD (CRF)

Conditional Random Field (CRF) unlike HMM, is a probabilistic framework that is not restricted by the Markov Independence assumption. Nevertheless, in theory, CRF models higher-order interactions between more than two states. In other words, it can model the interactions between the observation at the current state and its predecessor. That is from a theoretical point of view. However, in the literature, the existing CRFs are confined to the first-order dependencies between adjacent states.
Hunter et al [78] introduced a CRF for MM as an alternative to the classical HMM. The overall sophisticated methods model the spatial and temporal relationship as well as the classical HMM. However, authors integrate the driving behavior in addition to the vehicle speed. Using the same paradigm, Yang et al [98] characterize a CRF model for MM, an example of CRF is presented in Figure 2.8.

To verify the effectiveness of the model, the authors performed the MM on a dataset from Shanghai taxis. Even if the overall accuracy reached was significantly important, the CRF used considered only first-order dependencies between states. Therefore, CRFs share the same inability as the HMMs to take into account contextual information. In addition

Figure 2.8: A CRF for 3 GPS observations. The map on top illustrates the simplified situation of identifying road states and path states given GPS observations in the road network. This requires 5 random variables, $y_1 : \{r_1, r_2\}, y_2 : \{p_1, p_2, p_3\}, y_3 : \{r_3, r_4\}, y_4 : \{p_4, p_5\}, y_5 : \{r_5, r_6\}$, to build the CRF for map matching. Thus, nodes y_1, y_3, y_5 linking with observations (black circles) are point nodes while nodes y_2, y_4 are path nodes [98]

to that, a learning procedure is required for the CRF in order to model the interactions between these states, which makes the CRF easy to utilize but heavy to structure.

2.3.2.3/ WEIGHTED GRAPH TECHNIQUE (WGT)

More sophisticated techniques have been developed to take into account the spatial geometric and topological structures of the road network, in addition to the temporal/speed constraints. One of these techniques is referred as Weighted Graph Technique (WGT) [157]. In WGT the matching process is performed through a weighted candidate graph. Lou et al [58] introduced the st-matching algorithm (cf Figure 2.9) in which the WGT process is summarized as three steps:
(1) Candidate Preparation: in this step, the candidate graph is initialized. Similar to most MM techniques, the candidates are selected based on a radius of measurements from the estimated position (GNSS position).
(2) Spatial and Temporal Analysis: this step is composed of two components. First, the spatial component analysis, similar to HMM based method, an observation probability and a transition probability are emitted to each candidate. These two probabilities are inferred from a scoring function that takes into account the distance between the position and the candidate, in addition to the road topology. The second component is a temporal analysis in which the speed of the vehicle is compared with the typical speed constraints on each candidate path. The objective of the spatial and temporal analysis is to weigh edges in the graph.
(3) Result Matching: in this last step, the path is inferred based in the weighted graph constructed.

Globally, methods that fall into this category, share the same design as the one presented by Lou et al [58], they only differ in the scoring function in the spatial and temporal analy-

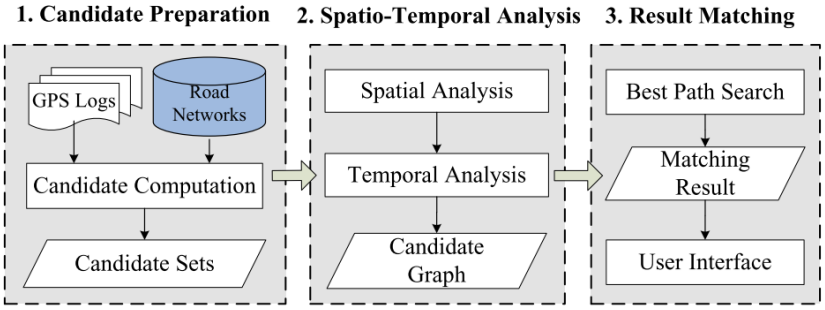


Figure 2.9: Overview of the st-matching algorithm [58]

sis. Thereupon, Hu et al [103] consider more sophisticated parameters. Indeed, authors take into consideration the reciprocal effects between adjacent candidates, the reasonableness of travel time, and other road characteristics (traffic lights, left turns, etc.).

2.3.2.4/ PARTICLE FILTER (PF)

Instead of dealing with an estimated position as an independent element, some researchers focus their attention on the trajectory of the vehicle. The idea is to filter this trajectory by coupling internal information from the microelectromechanical device (MEMS) such as gyroscopes and accelerometers, with GNSS. In essence, there exist two types of filters: linear and non-linear. For the linear type of filter, errors due to the imperfection of the model and sensors are represented by Gaussian white noises and are linearized using an approximation of a Taylor series. Based on the assumption of additive Gaussian white noises, the estimation of these error states can be obtained (for example using an Extended Kalman Filter(EKF)). In contrast, a non-linear filter does not require linearization, therefore, from a theoretical point of view, there are no errors resulting from the linearization step. In the context of MM, Dmitriev *et al.* [15] acknowledged that during a vehicle turn, the posterior distribution of the vehicle position on the road is non-Gaussian. Because of that, non-linear filtering methods are required to solve this problematic.
In order to tackle this non-linear problematic Particle filter (PF) is used. Initially, PF have been used as support prior to the MM process, by fusing sensors information to estimate the vehicle's state. For example, Toledo-Moreo *et al.* [60] propose a Lane-level localization MM (details we be discussed in Section 4.2.1) whereby he introduced a PF to fuse sensors information in order to estimate the vehicle a priori position. In general, the PF is structured as follows. In the initial phase, $N_{particle}$ particles are sampled. These particles represent the different locations of the vehicle and they got the same weight. For each particle, its weight is updating as soon as a new observation is received. Afterward, a resampling stage starts. Particles with low weights are likely to be erased, and the ones with higher weights are used in a vehicle cinematic model in order to feed particles of the next cycle. Therefore, all the methods that fall into this category share the same strategies [107], they differ in the definition of the weighting function for the particle.
One major drawback of these methods is that they employ a vehicle dynamical model (e.g., Ackerman model) that does not work for data that have a low sampling rate (e.g., over $5\ s$).

2.3.2.5/ MULTIPLE HYPOTHESIS TECHNIQUE (MHT)

The Multiple Hypothesis Technique (MHT), as the name suggests, holds a set of candidates or *hypotheses* during MM. The set of hypotheses is generally initialized based on a simple geometric metric. Afterward, the set of hypotheses keeps evolving as further observations are received. According to Kubicka *et al.* [134], the evolving process for MHT consists of two processes, namely, **hypothesis branching** and **hypothesis pruning**.
A hypothesis is branched (or replaced) when the vehicle travels the candidate and therefore arrives at a crossroad. The original parent hypothesis is then replaced by new child hypotheses. The new child hypotheses are an extension of the parent hypotheses by taking into account all the directions that the vehicle can take at the crossroad, which guarantees that there will at least one hypothesis covering the correct candidate in which the vehicle will travel. An example is illustrated in Figure 2.10. Another advantage of the method is that some failures are intuitively spotted. If there are no hypotheses, it necessarily implies that some problem has occurred. Hypothesis pruning consists of the elimination of the not acceptable hypothesis. The process is based on a pruning criterion. Leafing through the literature, this pruning criteria differ from one author to another. However, the main idea is to model criteria that allow to keep the most likely hypothesis and simultaneously eliminate the most unlikely hypothesis.

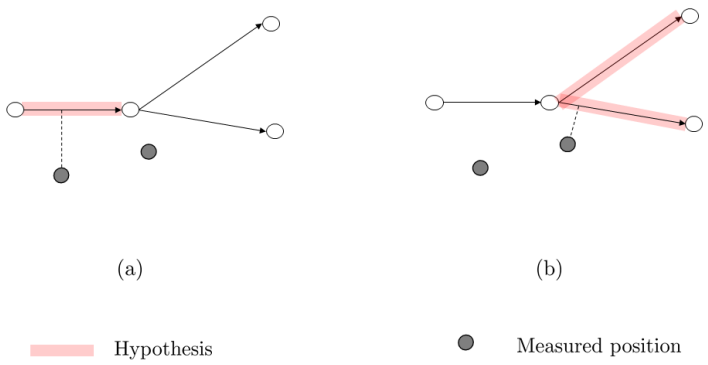


Figure 2.10: Principle hypothesis branching observed on two consecutive measured positions.

Historically, MHT was aimed for military tracking aircrafts like airplanes and missiles. Indeed, in 1979 Donald B. Reid, a U.S. Military engineer, developed a tracking multiple targets and multiple hypothesis algorithm [5]. The algorithm was adapted by Pyo *et al.* [19] into single target tracking with multiple hypotheses. The authors demonstrated that although the initial algorithm was designed for aircraft, it can easily be adapted to MM. The authors propose a MHT in which each hypothesis is associated with a probability. The pruning strategy consists of eliminating the hypothesis for which the probability goes under a certain defined threshold. The authors also consider the hypothesis with the highest probability and compared it to a predefined threshold to know if the hypothesis is confirmed. The presented method showed significant result with a range of 4% to 17% of a miss-match. However, this method is not optimal in the sense that optimal MM does not require any hand-tuning. Continuing on the same paradigm, Marchal *et al.* [26] and Kubička *et al.* [90] proposed their own version of the MHT. The contributions are essentially based on making the MHT simpler and faster. In addition to that, Kubička *et al.* [90]

presented a framework that does not require access to vehicle odometry and gyroscope data.
Compare to the HMM model, the MHT is more robust for miss matching since the current MM is not governed by transition state that is related to the previous solution. However, in worst-case scenarios, the set of hypotheses can grow exponentially, and therefore, the pruning process is critical. The pruning strategies have to be at a minimum robust to outlier hypothesis, but, at the same time, it has to be flexible in order to not eliminate the hypotheses that are likely to be correct. In that regard, the pruning process in a MHT is more prone to error, if not treated properly. On the other hand, there exists no formal proof for pruning strategies, or criteria, that ensure the latest condition. However, to overcome this issue, Quddus [33] introduced the notion of "integrity monitoring", which was inherited from aerial navigation where it used to verify the reliability of critical aerial navigation systems like satellites and missiles. In the context of MM, the need for "integrity monitoring" emerges from the fact that correct MM is not always possible when there are some strong ambiguities and some incongruities between the path and the map. In these situations, it is necessary to report the reliability of the MM output to the user. To this end, Jabbour *et al.* [44] followed by Li *et al.* [79] proposed different metrics (we invite the reader to their respective works from more details) to ensure that the output of the MM is coherent, and more importantly, to report false MM results.

2.4/ CONCLUSION

In the light of the investigated literature, we discuss what are the main requirements that are need to be necessary addressed by a MM algorithm. Despite more than twenty years of development on MM, there is not yet a solution that deals with all scenarios. To the best of the author's knowledge, there exists no consensus on how MM methods should be rated. Indeed, the majority of the authors utilize their own dataset. Nevertheless, according to the literature reviews [157] [134] the majority of the MM algorithm reaches an accuracy of around 95%. Accordingly, the accuracy of the MM will not be discussed in detail. Indeed, the major bottleneck in map-matching algorithms development is the absence of a unanimous agreement on one publicly accepted dataset. For that reason, the comparison between algorithms is difficult to conduct. One would like to bring to the intention of the reader that this problematic will be discussed later in the Chapter 5.
With that background we propose a different type of comparison, aspects considered in the depicted comparison are the most essential MM algorithm namely, uncertainty-proof, matching break, integrity indicator, and run time.
Uncertainty-Proof is the ability of the MM algorithm to take into account inherent uncertainties that come from the raw data.
Matching Break the MM algorithm has to be able to propose a solution in where there is a breaking in the GNSS data.
Integrity Indicator is a trust indicator on the validity of the output of the MM algorithm, which can be relevant for the ambiguous case.
Run Time in order to be used in an autonomous vehicle, the MM algorithm has to fulfill the real-time requirement.
Based on these criteria, we rate each method based on the previous section and we report the comment on Table 2.2. Further, this table summarized what has been already said in each part of this chapter. The notion goes from - -, -, 0,+, ++. With - - being the worst notation, and ++ being the best notation.

Table 2.2: Summary of MM algorithms in terms of uncertainty-proof, matching break, integrity indicator and run time.

Methods	**Matching Break**	**Uncertainty-Proof**	**Integrity Indicator**	**Run Time**
		DETERMINISTIC METHODS		
Geometric	- -	- -	- -	++
Pattern-Based	- -	- -	- -	++
		PROBABILISTIC METHODS		
Hidden Markov Model (HMM)	0	++	+	+
Conditional Random Field (CRF)	+	++	+	0
Particle filter (PF)	-	++	++	+
Weighted Graph Technique (WGT)	- -	++	+	+
Multiple Hypothesis Technique (MHT)	+	++	++	-

Consequently, deterministic methods are straightforward techniques that do not require complex computation. As a result, the running time of the algorithms is very low compared to other methods. However, these methods are very tributary to the quality of the raw data used. Therefore, they suffer from the incapability to handle uncertainties and ambiguities. On the other hand, probabilistic methods are analytically different from each other. Nevertheless, they have power in their capability to handle uncertainties and matching break situations. The backlash is that they are complex and require more computations.
In that sense, a hybrid architecture that includes a preprocessing step based on a deterministic method, and a selection stage based on a probabilistic method, has the utility of dealing with the limitations raised from each one of the methods. Indeed, the processing stage is straightforward and does not necessitate extra computations. Therefore, it has the strength to deal with the majority of the nominal case. On the other side, the ambiguous case can be solved using a probabilistic method, which has the ability to deal with these situations. By doing so, the complexity of the MM problematic is reduced. Following a detailed study of the state-of-the-art techniques and taking into account the scope of the thesis, we conclude that the Hidden Markov Model formalization of the problematic is the most relevant and the most suited for our use case scenarios. Further, if the good equilibrium between both approaches, deterministic and probabilistic, is found, we believe

that this can be the basis of a powerful MM algorithm.
In the next chapters, we investigate local localization methods which constitute among the main components of a localization architecture. The objective is to underline the main goal of this work, by presenting the most pioneering solution on that topic.

3

EGO-LANE LEVEL LOCALIZATION

Contents

3.1 Introduction 44
3.2 Related work 45
3.2.1 Model-driven approaches 46
3.2.2 Feature extraction 49
3.2.3 Fitting procedure 52
3.2.4 Learning approaches 54
3.3 Conclusion 58

3.1/ INTRODUCTION

For some applications like lane-keeping, knowing the road on which the vehicle is traveling is not sufficient. These systems must be informed about the position of the host lane in the road to provide adequate maneuver instruction and maintain vehicle safety. Further autonomous vehicle applications need a more accurate localization which can be translated by the knowledge of the lateral and longitudinal position of the vehicle in the ego-lane. For instance, overtaking maneuvers need a faultless knowledge of the lateral position of the ego-vehicle with respect to the ego-lane marking in order to decide whether the vehicle should overtake the obstacle or not.
J.mccall *et al.* [31] summarize the main objectives of lane position-detection algorithm systems as illustrated on Figure 3.1. The characteristics of these systems are given as follows [121]:
(1) Lane-Departure-Warning Systems: for this kind of system it is essential to accurately estimate the position of the vehicle with respect to the ego-lane marking.
(2) Adaptive Cruise Control: for this task, it is important to monitor the driver's attentiveness to the lane-keeping task. Measures such as the smoothness of the lane following are important for such monitoring tasks.
(3) Lane Keeping or centering: [130] the aim here is to keep or center the vehicle in its host lane. As a result, a faultless estimation of the lateral position is required.
(4) Lane Change Assist: [131] for this task, it is mandatory to know the position of the ego-vehicle in its host lane. On the other hand, the lane change has to be done without the danger of colliding with any object.

Under these considerations, in this chapter, we propose to classify the algorithmic techniques independently on the various modalities used (Camera, Lidar, Radar) in order to detect the ego-lane marking. Nevertheless, in the following, we will discuss some of the pertinent works related to the domain. Due to the vastity of the subject (we can write a thesis about lane marking detection) we will discuss only works that are relevant to us. Therefore, if some works are not quoted or neglected, the reader is urged to excuse these omissions.

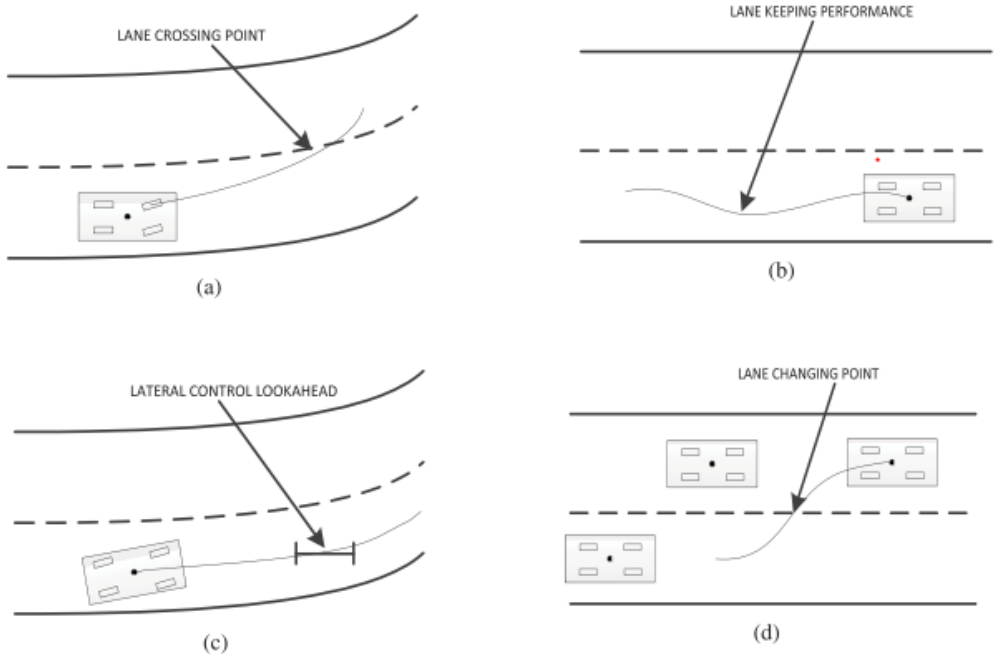


Figure 3.1: Some lane detection systems [11]: (a) Lane departure warning (b) Adaptive cruise control (c) Lane keeping or centering (d) Lane change assist. (image credit [121]).

3.2/ RELATED WORK

Leafing through the literature, it appears that most researchers use lane marking detection to provide an accurate ego-lane level localization. Analyzing this body of literature, lane marking detection has been an active field of research for the past three decades and frenetic progress made in the past few years.
It is possible to roughly categorize existing approaches to the lane marking detection into modular pipelines or model-driven and monolithic end-to-end learning approaches. As shown in Figure 3.2, both approaches are juxtaposed at a conceptual level. The standard approach to the lane marking detection is the model-driven approach. The main concept is to break down the lane marking detection into modules that can be independently developed and tested. Modular pipelines have the main advantage of deploying human-interpretable intermediate representations to understand system failure modes. A significant drawback to modular methods is that intermediate representations built by humans are not inherently suitable for tasks such as the identification of lane markers.

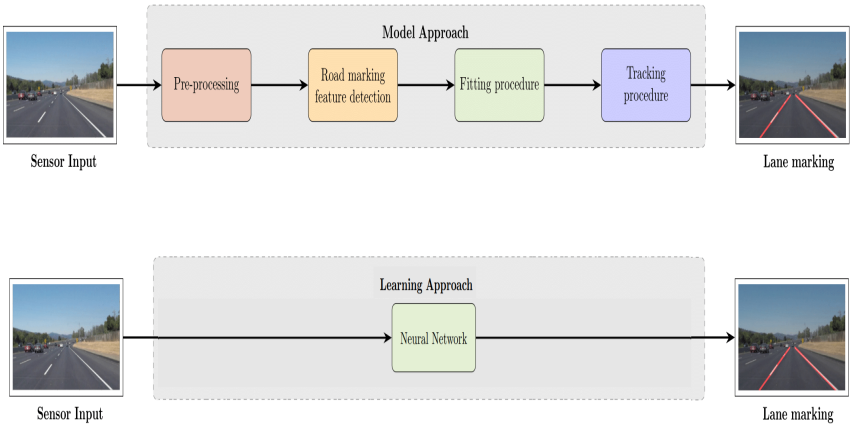


Figure 3.2: Model driven approach (top) vs monolithic learning approach (bottom)

An alternative to modular pipelines is end-to-end learning-based models based on neural networks. The network parameters can be learned via a training data-set or using a learning transfer technique from trained networks. These approaches reach significant accuracy in several computer science domains. However, most networks are trained and validated on one dataset, which makes the network less generalized to other datasets. Moreover, as claimed by Janai et al [161] neural network-based approaches are often hard to interpret as they present themselves as “black boxes” to the user which does not reveal *why* a certain error has occurred.

Although, several types of research have been conducted to review the current perception techniques for AVs [155]. The survey presented by Hillel *et al.* [89] is more adequate to the lane marking detection. The authors presented a review of ongoing research on road and lane detection. The review presented by the authors is widely cited and covers a large part of the lane marking technique used at that time, in addition to that, the authors presented the techniques without differentiating on the modality used. Nonetheless, this study was published in 2014, at a time where the dominance of neural networks was not as pronounced as it is now. Therefore, the study presented by Hillel *et al.* [89] is outdated but can still be used as good support to present the model-driven approaches. Concerning the learning approaches, there has been an enormous amount of effort invested in deep learning techniques for AVs and especially for the perception task. In that sense, the survey presented by Fayyad et al [159] covers most of the recent deep learning techniques.
Contrary to the study presented by Hillel *et al.* [89] and other noteworthy works presented in [31, 92], the focus and the ambition of this chapter are to emphasize the recent studies that dealt with lane marking detection, which included deep learning algorithm. On top of that, the chapter concludes by highlighting and summarizing some of the current trends approaches in lane marking detection, and a comparison between methods is given.

3.2.1/ MODEL-DRIVEN APPROACHES

Inspection of model-driven lane marking detection literature brings to light that most approaches share the same functional architecture. Thence, we depict the commonalities

between all the encountered algorithms into a generic system, whose components are divided into four steps, namely, a pre-processing step, succeeded by a road marking feature detection, then a fitting procedure, and finally a tracking procedure. An illustration of these components are illustrated in the Figure 3.2 (top). We use this generic system as a skeleton enabling comparison between different algorithms according to their functional parts. Naturally, feedback connections also exist between higher modules (e.g fitting procedure) that guide lower module (e.g, pre-processing).

3.2.1.1/ PRE-PROCESSING

In general, a set of several operations that can be applied to a frame input (we use the word frame, to embrace both lidar and camera image) before feature extraction is called pre-processing. In general, it is the first process of lane marking detection. The objectives of pre-processing are to enhance features of interest, reduce clutter, and remove misleading artifacts. Thereafter, the cleaned image is used for feature extraction. According to [89] the methods that fall under this module's scope can be clustered into two classes: handling of lighting-related effects and pruning of non-relevant or misleading parts of the image.
A robust lane marking should be capable of handling different lighting conditions which are constantly changing because of the effects of time of day and weather conditions. These lighting conditions can vary from sunny midday to nighttime artificial illumination. In addition to these natural changes, the lane marking solution might be confronted with a drastic lighting switch when entering or exiting a tunnel, or while being under a bridge that covers the lighting suns. Another illumination phenomenon that has to be brought seriously to the table is the illumination of the lens flare caused by the sun's rays hitting the camera's Field Of View (FEV). Ergo, Huang *et al.* [54] use the "absolute calibration" of the camera allows the computation of solar ephemeris, which allows deducing the sun location on the image and hence suppression of the line estimates that point toward the sun location on the image (Cf, Figure 3.3).

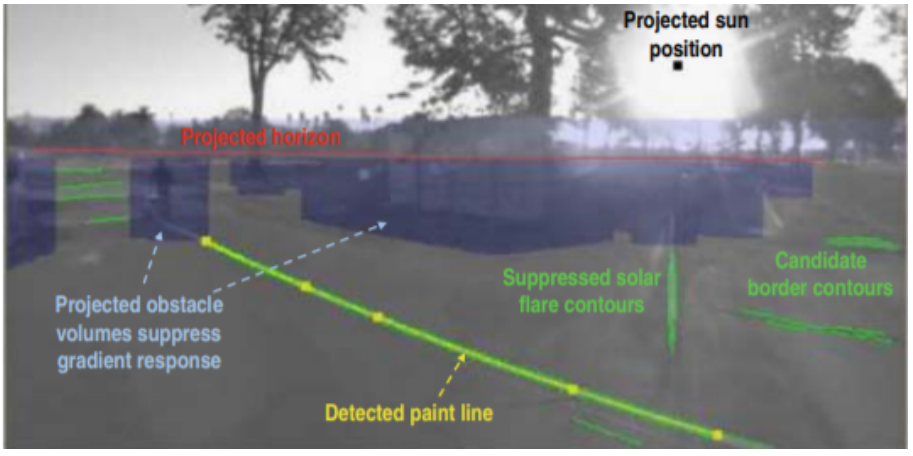


Figure 3.3: Use of absolute camera calibration to project real-world quantities, such as the position of the sun on the image (credit [54]).

Furthermore, a major source of clutter is the shadows cast on the surface of the road.

Their edge intensity can be ambiguous for some gradient feature extractors. To circumvent this illumination-related issue, several researchers tackled the subject and propose several solutions. In a global manner, color-space transformations are performed on the image. To name but a few: Hue Saturation Lightness (HSL), Lightness and A and B (LAB), and The Luma component (Y), the Blue-difference and Red-difference Chroma Components (YCbCr). The General assumption made for these color manipulations is that hue information does not change in the shadowed region of the image, implying that the hue information is not affected by the level of illumination of the image, or that the effects can be compensated [29, 38, 56].
The second category of image pre-processing techniques includes the pruning of non-relevant or misleading parts of the input sensor that can miss-lead the lane detection process. The apparent difficulty of the methods that fall into this category is to insulate the artifacts in the image sensor. To archive this objective, many studied have been developed over the years. In that context, objects like cars and pedestrians are treated like obstacles for lane detection. Therefore, several techniques have been followed in order to detect them and remove them. Hence, Huang *et al.* [54] enhance the detection process with $3D$ data from lidar, the lidar cloud points facilitating rapid removal and rejection of off-ground points that are considered as obstacles. One can notice that differentiation between obstacles is not made, a similar approach was presented in [53, 148]. Using $2D$ flow of image points, Yamaguchi *et al.* [51] propose a method based on the alignment of two successive images. The Structure-From-Motion technique was applied to infer the road region depending on the image motion (Cf, Figure 3.4). However, these techniques were later on abounded due to high false-positive rate [41].

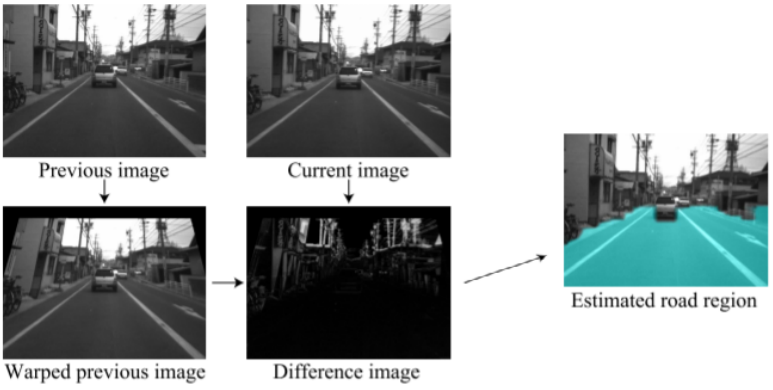


Figure 3.4: The pre-process fro eliminating non-relevant part of the image. (credit [51]).

Furthermore, the most utilized approach for the punning part consists in defining Region Of Interests (ROI). Hence, only these regions will be focused on the feature extraction. Thus, several works have addressed the issue of how the determine these ROI. The most simple technique consists of taking the lower half of the image as ROI [61]. The naivety of these techniques makes it very limited. Therefore, other researchers use the correlation between the 3D world model and 2D image in order to delineate the ROI. To achieve this goal, it is required to know the camera pose with respect to the ground surface. Huang *et al.* [54] claimed that the correlation is constant and hence the calibration was made beforehand. The same idea is shared by Nava *et al.*, [136], the authors propose a ROI delimitation using the vanishing point of the road. But, as claimed by the authors, this

method does hold only for small roll angle conditions. However, this strong assumption does not hold for the curved road. Wherefore, Aufrère *et al.*[18] propose a probabilistic road model that links between the geometric of the road, and the uncertainties about the camera pose. By doing so, the uncertainties about the model define the ROI in the image (cf, Figure 3.5). This idea was extended in [50]. In the same manner, authors in [11, 61] use the depth computation. This computation allows to determine the vanishing point on the image and hereby the delimitation of the ROI. As claimed before, feedback

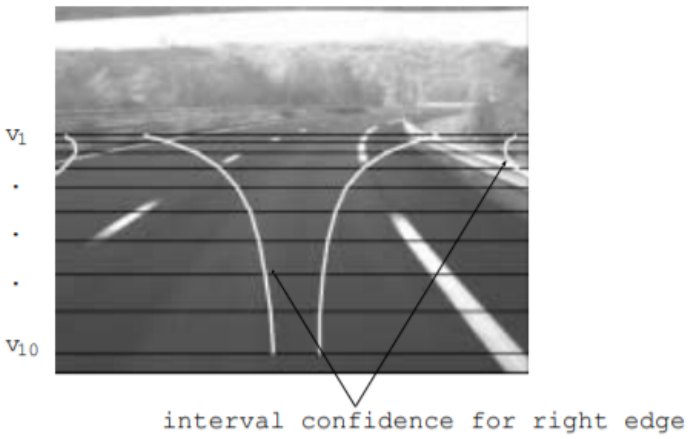


Figure 3.5: ROI definition using the probabilistic model. (Image credit [18]).

connections do exist between the higher functional block and lower block. To illustrate that, an interesting approach is presented by Alvarez *et al.* [83]. It relies on the use of road priors and contextual information coming from a digital map in order to determine the shape of the road in the image. Depending on the number of lanes and the width of the lane, the road skeleton is built and smoothed using cubic interpolation. The retrieved road skeleton is then projected onto the Bird' Eye View (BEV) image by taking into account uncertainties related to the vehicle's pose. The whole process is shown in Figure 3.6.

Contrary to the previous methods, Caceres *et al.*[100] emphasize on the collision risk region, which is extracted taking into account the vehicle speed, and therefore, the ROI size increases as the speed increases and vice versa. An illustration of the idea is given in the Figure 3.7.

3.2.2/ FEATURE EXTRACTION

Once the irrelevant part of the input sensor is punned, the remaining relevant part of the input sensor is supposed to contain some pieces of the lane marking. These pieces gathered together should contain all the necessary information needed in order to fit the lane marking. Throughout this work, these pieces are often called primitives or features. Indeed, feature extraction is a crucial step of lane marking recognition. Hence, in the majority of the works that fall in the model-driven approach, the fitting procedure is tributary of the outputs of the feature extraction. Therefore, in case of a momentary failure for the feature extraction, recovery becomes almost impossible.
Study this corpus of literature, reveals that the majority of the approaches presented rely

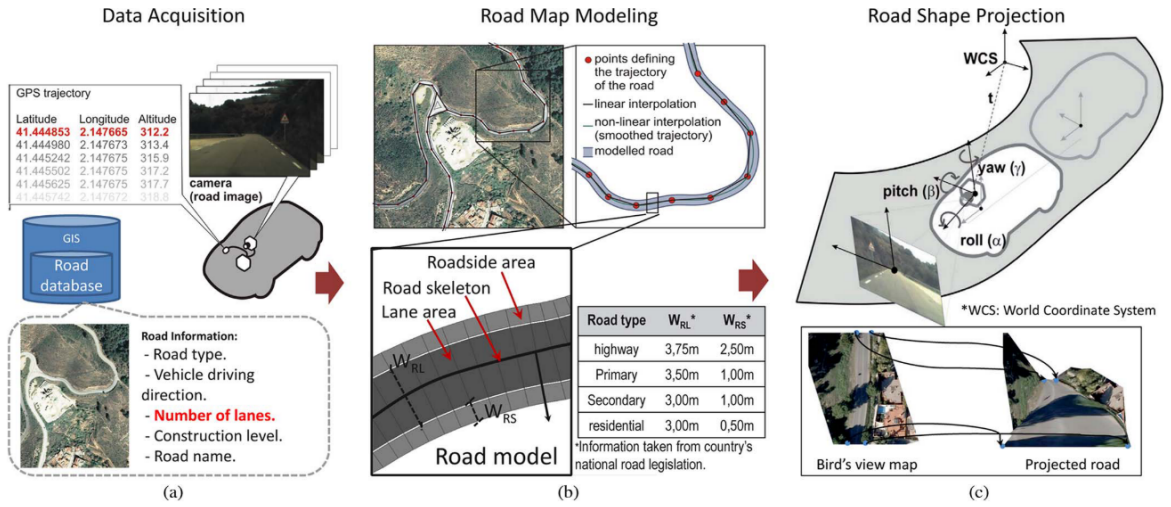


Figure 3.6: Proposed algorithm to estimate road priors using a digital map. (a) Acquisition of the geometrical information from the digital map database. (b) The modeling of the road depending on the acquired information. (c) Road shape projection in the BEV (credit [83]

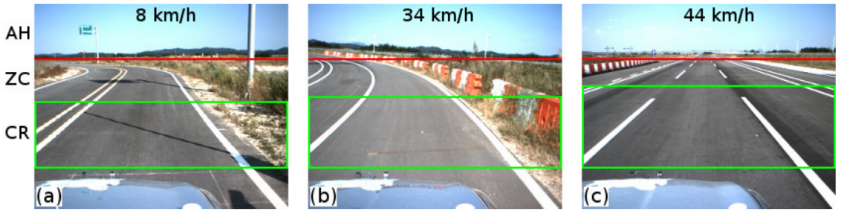


Figure 3.7: ROI estimation (in green) based on the vehicle speed.(Image credit [100]).

on bottom-up feature extraction. Indeed, lane markings are easily discriminated by shape and color in the image and it is possible to determine whether a lidar beam has intercepted asphalt or road painting regardless of the lighting conditions [87].
Lane marks can be detected either based on their shape, color, or their combination [89]. To that end, several strategies have sprung up. The main assumption made is that lane markings are distinguished by their appearance from the rest of the road surface. This assumption leads to a whole family of filters based on gradient detection [18, 34, 35, 47]. However, these techniques can be time-consuming if there exists noise caused by different types of markers (e.g, arrows in the middle of the lane). To reduce this kind of noise, researchers worked on solutions that filter the edges that are not in the vertical direction. These filters are known as steerable filters [31]. Steerable filters enable to follow the orientation's change along the lanes marking in the image by convolution with only three kernels. Following the same spirit, several hand-crafted filters were proposed to extract fragments of lane marking, namely, Sobel filters [23], Statistical Hough Transform (SHT) [64], top-hat filter [132], and histogram-based filter [16]. A comparison of theses features extractors have been presented by Veit *et al.* [49], the authors applied different feature extraction on the same input as shown on figure 3.8.

Regardless of the type of gradient filter used, the kernel of the filter has to be adjusted before applying it to the image. However, the perspective distortion of the camera makes these adjustments not suitable for the entire image. To bypass this problem, a com-

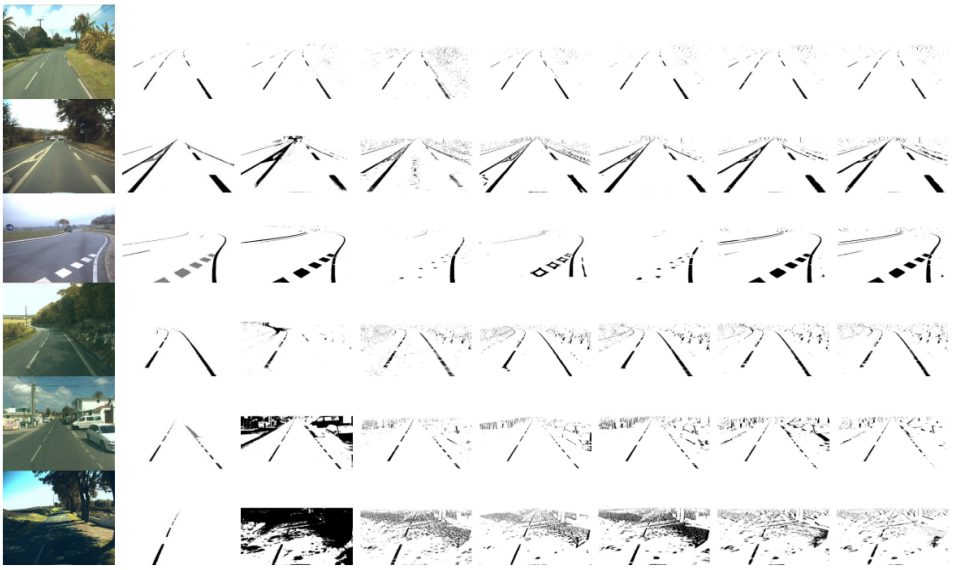
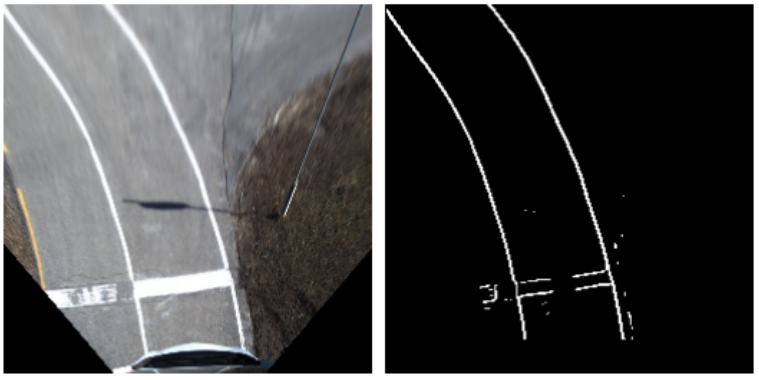


Figure 3.8: Comparison of different features extractors. The first column is the original image. The second column is for ground truth. Third-eighth columns (from left to right): global threshold, + - gradients, steerable filters, top-hat filter, local threshold and symmetrical local threshold (image credit [49])

mon approach is to transform the image into another perspective called the "inverse-perspective" [28, 31, 35, 46, 52] image called the bird's-eye view (an example is giving in the Figure 3.9). In the bird's-eye view, the width of the lane markings is equidistant. This property has a lot of advantages, for example, it is a convenient common space to fuse multiple sensors information. Moreover, the representation facilitates some fitting produce that will be discussed below. However, these advantages come with a cost in terms of computational time and loss of resolution.



(a) Bird's Eye View (b) Steerable Filters

Figure 3.9: Result of a BEV image (credit [142]).

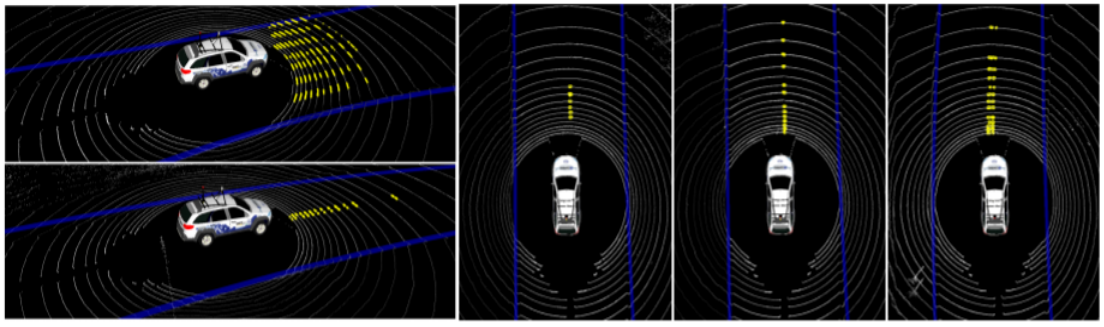
When dealing with lidar data, the main assumption made is that lane marking can be distinguished based on their reflectivity. Therefore, authors in [32] present a threshold in order to discriminate lane marking. This approach was extended in the work [88]. The authors presented an Otsu thresholding method that separates the lidar point clouds into

asphalt and road marking (Cf Figure 3.10).

3.2.3/ FITTING PROCEDURE

Going down the timeline, preliminary works on lane detection focus on highway scenarios where the curvature's change is small enough to be neglected. In that context, pioneering work on the road model was initially proposed by the group of Prof. Dickmans *et al.* [7] in 1986. The latter presumes to build a mathematical 2D model that describes the lane geometries, yielding to a high-level representation of the lane marking with the use of a clothoid road model for planar roads. Buoyed by this successful achievement, the group of Prof. Dickmans extended the approach by using a 3D model lane representation that included a clothoid parameter and a curvature in the vertical direction [13]. These works lay the foundations of several works with the objective to know what kind of lane model can represent accurately the lane marking and which fitting strategies should be adopted consequently.
By and large, the main objective of this fitting procedure is to extract and high-level representation of the path. This high-level representation is the sine qua non to a higher block of AVs like decision making and control. Thereby, the choice of the type of lane model is crucial.
In practice, when fitting a model to noisy data there exists a compromise between over-restrained models that do not tolerate all the existing geometries and under-restrained models that tend to over-fit on noisy features. Scrutinizing this body of the literature, the lane model can be clustered into three heterogeneous modeling techniques, namely, parametric, semi-parametric, and non-parametric. Such classification allows a clear understanding of the distinctiveness of certain techniques. Furthermore, the assumptions made for each category help to understand some failures of some of these techniques.

- **Parametric model:** methods that fall into this category made the strong assumption of a global lane shape (e.g lines, curves, parabola). These models fall when dealing with non-linear road and lane topologies (merging, splitting, and ending lanes). Indeed, the geometric restriction imposed by the parametric model does not tolerate such scenarios. Concerning the fit strategies used, several regres-

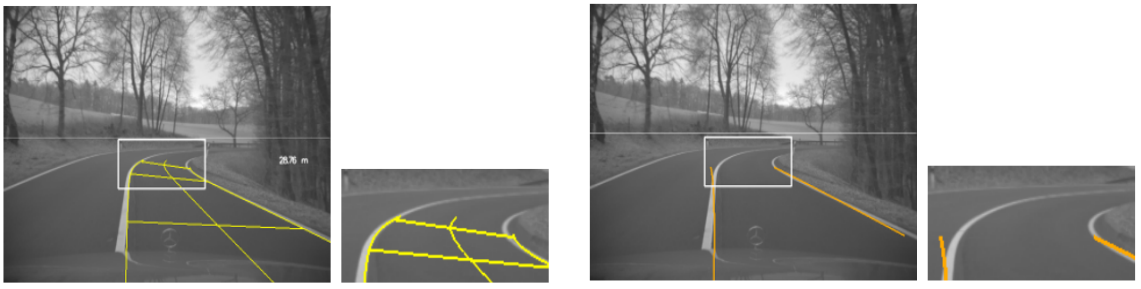


(a) Top: crosswalk; Bottom: double line (b) Dashed line (c) Continuous line (d) Double line

Figure 3.10: Feature extraction through a modified version of Otsu method. Blue lines are the road bounds estimated by curb detector and yellow points correspond to the detected road marking. (Image credit [87]).

sion techniques have been used (e.g RANSAC, least-squares optimization, Hough transform, Kalman filter)

ii **Semi-parametric model:** contrary to the parametric model, semi-parametric model do not assume a specific global geometry of the road. On the downside, the fitting model can over-fit or have unrealistic path curvature. The lane marking is parametrized by several control points. Different spline models with different control points have been used (e.g, Spline, B-spline, Cubic spline). The appearing complicatedness of these models is in choosing the best control points. Indeed, the number of these points affects the curve complexity. In addition to that, these points should be homogeneously distributed along the curve of the lane marking in order to prevent unrealistic curves (Cf Figure 3.11).



(a) 3D B-Spline model (b) Planar clothoid model

Figure 3.11: Two lane marking models on the same rural scenario. Images (a) present a 3D B-spline model and images (b) illustrates a planar clothoid model(image credit [65]).

iii **Non-parametric model** it is the less conventional approach. The main prerequisite needed is continuous but not necessary differentiable. This model has more freedom to model the lane marking. Meanwhile, it is more prone to erroneous modeling leading to unrealistic curves.

In the following, a list of works regarding the three categories is presented in table 3.1

3.2.3.1/ TRACKING PROCEDURE

The vast majority of lane marking detection system integrates a tracking mechanics that integrate knowledge from the previous frame to improve the knowledge on the present frame. According to Hillel *et al.* [89], this mechanics has three major goals: improving the accuracy of correct detection, reducing the required computation, and correcting erroneous detections. A tracking procedure can be used in a lane marking detection system in two different modes: using the detection results from the previous frame, the tracking system can enable the definition of ROI in the current frame. By doing so it will reduce the size of ROI. Aufrère et al [18] used the vehicle's information in order to update a probabilistic model, which contains all the ROI delimitation. To do so, authors integrate these proprioceptives information in an Extended Kalman Filter (EKF). The same idea was adopted by Wu *et al.*[50]. These approaches can successfully decrease the signal-to-noise ratio by updating the ROI after each iteration. However, it should be noticed that the main assumption made is that the model is capable of capturing the motion between

Categories	Geometric methods	Fitting method	Advantages	Disadvantages	References
Parametric	Straight lines	Hough transform and its variants	Straightforward approach shows good approximation for short range lane marking and can be valid in highway scenarios	Unfit for curves roads which is the cases in most rural roads	[11, 27, 28, 52, 57, 61].
	Polynomial model	RANSAC, least squares optimization	The spectrum of application is greater than the linear model. In addition Polynomial models has the ability to estimate the parameters of the road.	Can not handle roads abrupt change of curvature. In addition, the geometrical assumption made are not always correct (e.g, taking 3-3.5m as a width lane)	[18, 54]
	Cloithoid	Extended Kalman filter	Can handle the situation where there is a abrupt change of the steering angle. For example, between the junction of straight and curved road.	The clothoid model is generally made of some simplifications in order to get a viable model	[22, 63]
Semi-parametric	The whole family of Spline including: Bspline, Bspline 3D, Catmull-Rom spline, and Cubic Hermit Spline	Energy-based optimization.	Capable of dealing with a large range of curved road using control points if accurately chosen.	The inconvenience of this model appears in the choice of the control points. Undoubtedly, the position of these control points will affect the general curve of the lane. A wrong choice of theses control points leads to unrealistic road shape.	[45, 51]
Non-parametric	Isolated points	Particle filter	The model is not governed by geometric restrains, which allows it to model more challenging road lane marking.	With no geometric restrains imposed, the fitted model can leads to unrealistic road model.Indeed, geometric correlation between lane marking are not considered.	[55]

Table 3.1: Various lane fitting models presented in the litterature.

two consecutive frames. At the same time, they assume that the latest detected lane marking is correct. Hence, no recovery strategy is made.
Alternative approaches consist of fusing lane marking detection and a digital map of the road that contains the position of the left and right lanes.

3.2.4/ LEARNING APPROACHES

One of the major challenges in estimating the lane marking is the need to have an accurate model that fits the features detected. Unfortunately, providing an accurate model that covers every road scenario is a complex task due to the singularity of some road scenarios (i.e, merging, splitting, and ending lanes). Moreover, inherent uncertainties coming from sensor data can not be mathematically modeled in the model-driven pipeline system. Along these lines, monolithic end-to-end learning approaches have the advantage to abstract the mathematical modeling for each functional block as presented in Figure 3.2. As a consequence, learning approaches if treated properly, are a powerful tool in order to correctly detect lane marking detection.
On that subject, several studies have been deployed. In their study, Kim et al [45] reviewed the performances of classical machine learning methods for features extraction, namely, Artificial Neural Networks (ANN), Naive Bayesian Classifiers (NBC), and Support Vector Machines (SVM). Nevertheless, the fitting model was still performed using the Spline model. On the same topic, Golan *et al.* [71] introduced a learning-based approach

based on a boosting algorithm to detect lane markings without requiring a predefined road model as illustrated in Figure 3.12. The algorithm was validated on several data collected on daytime and night-time proving that the classical machine learning method can be useful for lane detection marking.

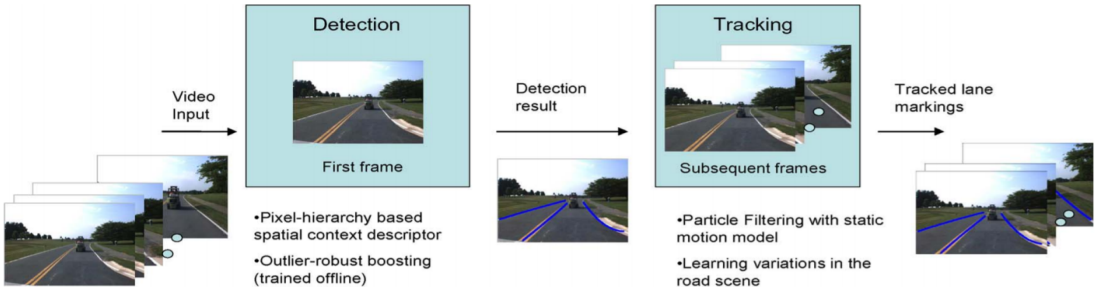


Figure 3.12: The main pipeline of the proposed Algorithm of Golan et al [71].(credit [71]).

Deep Learning (DL) is the natural upgrading of classical machine learning. DL aims to imitate the function of the most successful machine, which is the human brain. Contrary to the classical machine learning methods, some of DL techniques do not require a motion-based model. Indeed, the networks are capable of excellent prediction. Historically, the concept of deep learning was founded in 1943 by Walter Pitts and Warren McCulloch [2]. However, it was until the 2010s with the development of powerful computing machines with the arrival of Graphics Processing Units (GPUs) combined with the availability of the "big data" needed to train the models that ignite the emergence of DL techniques. These days, the scope of application of DL ranges from to finance [120], healthcare(U-net [97]), and self-driving vehicles [123].
Going back to AVs, several DL algorithms exist that are used for several applications depending on the properties of the network. However, concerning the lane marking detection, there are two major Deep Learning (DL) architectures that draw the attention: Convolutional Neural Network (CNN) and Recurrent Neural Networks (RNN). Going back to the time, CNN gained notorious popularity in 2012 when Krizhevsky *et al.*proposed AlexNet [72] and won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). A dissection of a CNN reveals that the network is composed of the following layers, as illustrated in the Figure 3.13:

1. Input layer: this layer contain the input data, for example, an image.

2. Convolution layers: in these layers convolutions operations are performed on the image

3. Pooling layers: the objective of these layers is to reduce the size of the output from the convolution layers.

4. Fully connected layer: classifies by connecting all the neurons

5. Output layer: the output of the network, in a CNN it is generally a classification probability

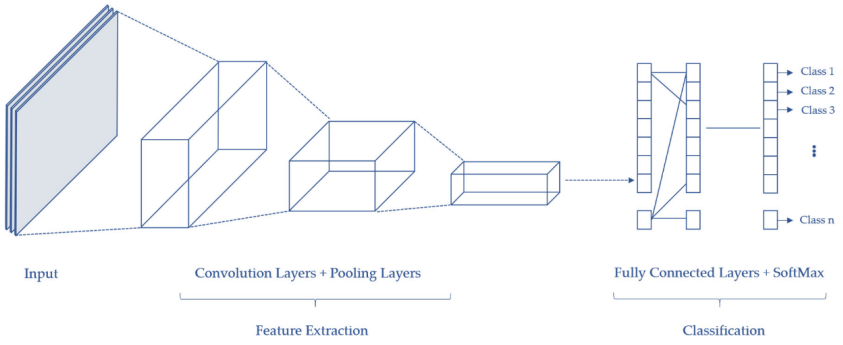


Figure 3.13: The general pipeline of a CNN.(credit [159]).

Following the success of CNN in computer science, researchers considered its uses to tackle lane detection. In that regard, Huaval *et al.*[96] were the first to use DL technique to identify pixel locations of the single lane on highways. Their model is based on the OverFeat (a CNN developed by team of Yann Lecun[80]). They trained their CNN on a private collected data-set on a highway in San Francisco (USA). The ground truth labels were generated using a camera, Lidar, Radar, GPS, and human annotations. The network showed successful results in terms of lane detection. But, the CNN was restricted to the detection of only the ego-lane on which the vehicle was traveling. Indeed, the general assumption made is that the vehicle is always traveling in the center of the lane. Building on this success, He *et al.*, [102] proposed a Dual-View Convolutional Neutral Network (DVCNN) framework for lane detection as illustrated in Figure 3.14. The input of the network is pre-processed (by a kind of hat filter), then front view and BEV images and the output of the network are line probabilities which are fed to an optimizer in order to find the right lane marking. The network shows promising results. However, the method required image pre-processing and post-processing.

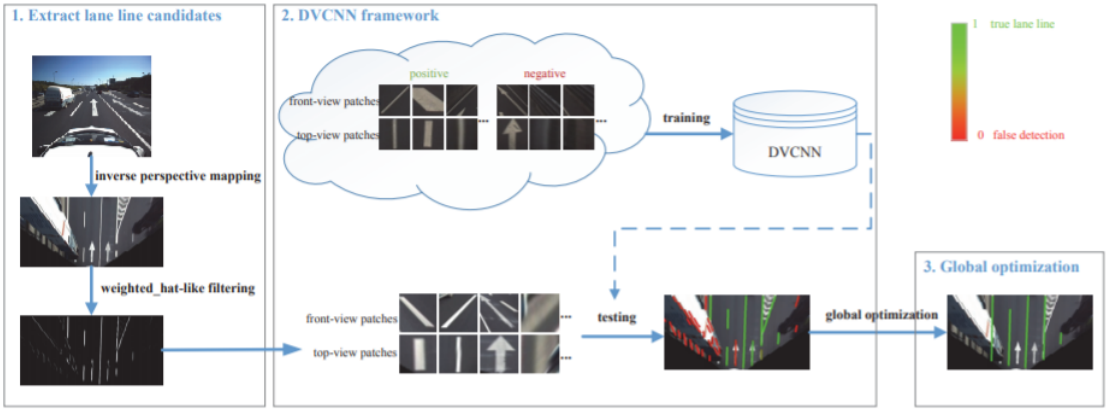


Figure 3.14: The general pipeline of the DVCNN(Image credit [102]).

Companies showed interest in this problematic. In that regard, researchers from Ford released an end-to-end framework called DeepLanes [101]. Unlike most of the works, the network detected lanes based on images coming from two laterally-mounted cameras looking downward onto the lane markers (as shown in Figure 3.15). Notwithstanding the good results obtained, the network was not widely adopted due to the fact that the network

was trained on a private database.

Figure 3.15: The setup of the laterally-mounted camera looking downward onto the lane marking(credit [102]).

Reproducibility, that was the watchword on the lane detection challenge that was held in CVPR'17 and in which the TuSimple database was released. The winner of the challenge was Pan *et al.*, [138] with the Spatial Convolutional Neural Network (SCNN), a Deep Learning technique designed for lane marking detection in traffic scene. The structure of the SCNN allows it to exploits the spatial information in the image. On the same topic, authors in [149] proposed a DL network called Line-CNN. The network had slightly better results than the SCNN, authors adapted the strategy of the regional proposal network (that exists in Yolo [104], for example) to the lane marking detection, they also trained their network with an additional dataset which was not publicly proposed. A succession of neural networks has been released after that. However as pointed out by authors in [169], many of the published works on DL for lane marking detection do not share their code in order to reproduce their results [149, 151, 172] therefore it hinders the comparison. In some cases, their code is only partially public, like the works in [160], and in worst cases, they do share their code but the reported results are not reproducible (ENet-SAD [145] [1]). The detection of lane marking using deep learning is an ongoing research topic, with multiple networks released every year dealing with the subject (it would not be a surprise if other deep learning methods would be released by the end of writing this manuscript). The majority of these algorithms are benchmarked. Currently, the Tusimple benchmark (see Table 3.2 for more details) is saturated with high values in accuracy and F1 score (see appendix for metrics definition), which is explained by the fact that the dataset is not complex and the metric permissive [169]. In that regard, a database called CULane was released. The objective of this database is to offer a more complex and more large public available dataset for lane detection. Considering the abundance of literature on the subject, tracking every deep learning algorithm proposed for lane detection is beyond the bounds of possibility. However, a site web called https://paperswithcode.com/ reviews all the lane marking detection methods that has been benchmarked on either Tusimple or Culane. In the Table 3.2 and the Table 3.3 it is given an overview of these methods, with a ranking based on their performances on both benchmarks. The algorithms presented are the ones with an available code.

[1] The explanation coming from the authors is that the difference in results is due to some engineering tricks [169]. However, this was reported neither in the code or in the published work

3.3/ CONCLUSION

This chapter discussed the major algorithms for ELL. We emphasize methods that are based on the detection of the ego-lane marking. Accordingly, model-based approaches have a strong ability to detect the ego-lane marking in various scenarios. The sequential pipeline of these methods allows a better partition of the ego-lane marking detection task into blocks. Each block is responsible for a specific task. Therefore, the formalization of the whole problematic is enhanced and the intermediate block representations allow a better system failure identification. Furthermore, they have a systematic modular architecture that enables them to improve or incorporate new functionalities that were not supported in the initial design without requiring significant modifications. Regardless of the method used, a fitting model is required in order to fit the detected feature to a predefined model. As a consequence, the generalization of such methods is complicated and challenging for highly complex road scenarios. Nevertheless, as claimed by Hillel *et al.*, [89], the model-based method does perform well in the majority of road scenarios.
On the other side, the monolithic learning approach based on Neural Network reaches a better accuracy for the detection of lane marking as the majority of benchmarks' leaders are deep learning methods (Table 3.2 and Table 3.3). Besides, they perform well when a model can not be formalized or is not available. The learning approach requires a learning phase that is performed on an annotated dataset. The procedure is performed outline, preliminary to the deployment of the network. However, in real-world applications, data is limited in quantity and is usually gathered for a specific task and for a specific configuration (e.g, same city, same camera). Therefore a change between the dataset and the validation test leads to a decline in the accuracy obtained. In addition, the learning procedure is time-consuming and the output can not be predicted in advance. Nonetheless, the major drawback of the methods based on this paradigm is that the network-based algorithm is hard to interpret as it is presented by "black boxes" to the user, which does not reveal why a certain error has occurred [161].
Taking into consideration these limitations, we believe that model-based is the most suited approach in order to perform the ego-lane detection with the objective to locate the autonomous car in his own lane for mainly two reasons. Indeed, while the deep-learning techniques for line detection are limited to image mask retrieval and do not provide a usable representation in world coordinates, model-based fits a 3D model in order to match the lane markings in the road. This feature is highly relevant notably in highway scenarios, where the shape of the road and the lane marking, are not randomly modeled. The second reason is the immutability to change of the deep-learning methods, which does not allows modification and amelioration by adding new features without changing the whole architecture of the network and retraining the network on adapted dataset. With that background and with relation to the previous chapter 2, we believe that a model-driven method can be enhanced by using a map to provide priors to the detection module and hence reduce the complexity of the computation.
In the next chapter, we will discuss the later block of the algorithm localization which is the ELL. In the following chapter, a review of the literature is given by presenting the most relevant works and the most promising techniques. To conclude, a discussion will be held in relation to our thesis work.

Table 3.2: Summary of DL algorithms benchmarked in Tusimple in terms of accuracy and F1 score

Models	**Accuracy**	**F1 score**	**Extra training data**	**Paper title**
PINet	96.75%	97.20	No	Key points estimation and point instance segmentation approach for lane detection [166] (2020)
ENet-SAD	96.64%	95.92	No	Learning lightweight lane detection cnns by self attention distillation [145](2019)
HarD-SP	96.58%	96.38	No	Towards Lightweight Lane Detection by Optimizing Spatial Embedding [162](2020)
Spatial CNN	96.53%	95.97	Yes	Spatial as deep: Spatial cnn for traffic scene understanding [138](2018)
Pairwise pixel supervision + FCN	96.50%	94.31	No	Learning to cluster for proposal-free instance segmentation [128](2018)
EL-GAN	96.40%	96.26	No	El-gan: Embedding loss driven generative adversarial networks for lane detection [126](2018)
LaneNet	96.40%	94.80	No	Towards end-to-end lane detection: an instance segmentation approach [137](2018)
ENet-Label	96.29%	95.23	No	Agnostic lane detection [144]
R-34-E2E	96.22%	96.58	No	End-to-end lane marker detection via row-wise classification [172](2020)
R-50-E2E	96.11%	96.37	No	End-to-end lane marker detection via row-wise classification [172](2020)
LaneATT (ResNet-122)	96.10%	96.06	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
ERF-E2E	96.02%	96.25	No	End-to-end lane marker detection via row-wise classification [172](2020)
LaneATT (ResNet-34)	95.63%	96.77	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
LaneATT (ResNet-18)	95.57%	96.71	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
End-to-end ERFNet	95.24%	90.82	No	Lane detection and classification using cascaded CNNs [152](2019)
PolyLaneNet	93.36%	90.62	No	PolyLaneNet: Lane estimation via deep polynomial regression [169](2019)

Table 3.3: Summary of DL algorithms benchmarked in Tusimple in terms of F1 score, the results showed stands for the total of all classes of the Culane

Models	**F1 score**	**Extra training data**	**Paper title**
LaneATT (ResNet-122)	77.02	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
LaneATT (ResNet-34)	76.68	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
LaneATT (ResNet-18)	75.13	No	Keep your Eyes on the Lane: Attention-guided Lane Detection(2020) [170]
PINet	74.40	No	Key points estimation and point instance segmentation approach for lane detection [166] (2020)
ERFNet-E2E	74.00	No	End-to-end lane marker detection via row-wise classification [172](2020)
CurveLane-M	73.50	No	CurveLane-NAS: Unifying Lane-Sensitive Architecture Search and Adaptive Point Blendin [171](2020)
ERFNet-IntRA-KD	72.4	No	Inter-region affinity distillation for road marking segmentation [160](2020)
ResNet34-FAST	72.3	No	Ultra fast structure-aware deep lane detection [168](2020)
ResNet-101-E2E	71.90	No	End-to-end lane marker detection via row-wise classification [172](2020)
Spatial CNN	71.60	Yes	Spatial as deep: Spatial cnn for traffic scene understanding [138](2018)
ENet-SAD	70.80	No	Learning lightweight lane detection cnns by self attention distillation [145](2019)
ENet-Label	68.80	No	Agnostic lane detection [144](2019)
ResNet18-UFAST	68.40	No	Ultra fast structure-aware deep lane detection [168](2020)

4

LANE LEVEL LOCALIZATION

Contents

4.1 Introduction 61
4.2 Related Work 62
4.2.1 Map aided approaches 62
4.2.2 Landmarks approaches 65
4.3 Conclusion 67

4.1/ INTRODUCTION

A fundamental aspect of a fully autonomous vehicle is its ability to properly evaluate the situation of the ego-vehicle with regards to the road environment. Part of this situation evaluation is the knowledge about some keys localization level components. In previous chapters, we presented two keys levels needed, to fully fit this fundamental evaluation: the road level localization, or the knowledge of which road the vehicle is traveling, and the ego-lane level localization, or the knowledge of the position of the ego-vehicle in its lane. For the majority of the Advanced Driver-Assistance Systems (ADAS) presented in Figure 3.1, a partial understating of the observed road scene is sufficient. Hence, the knowledge of these two components is sufficient. However, the lane/road understanding demands in terms of precision and false alarm rate [89] vary from one application to another. Therefore, for some tactical higher-level intelligent safety systems, the knowledge of the lane on which the vehicle is traveling is critical. Indeed, knowing the host lane can serve the autonomous navigation system in providing the most adequate instructions depending on this lane, for example, it can provide a better overtaking strategy.
In the broadest sense, Lane-Level localization is a meaningful concept that can refer to two distinct topics: it refers to the determination of the ego-lane, which means the lane on which currently the vehicle is traveling. At the same time, it may refer to the estimation of the lateral position of the vehicle inside the whole road. While the latter definition is an estimation problematic and solutions live in $\mathbb{R}$, the first one can be interpreted as a classification problem and its solutions live in $\mathbb{N}$. The two paradigms lead to the same knowledge, which is the LLL or the localization of the host lane.
There exist abundant systems that can help the AVs to obtain the LLL: using a GNSS receiver to locate the ego-vehicle in the road. The lack of accuracy provided by a classic

GNSS that can be caused by poor satellite signals, high degree dilution of precision, or multi-path in urban scenes, is first compensated with proprioceptive sensors, such as the Inertial Measurement Unit (IMU). These methods are well known as Dead Reckoning [73]. Therefore, the position of the host lane is inferred by combining the obtained localization with a coarse digital map. Unfortunately, the accuracy of the localization obtained is in the order of meters. Indeed, according to the Federal Aviation Administration (FAA) GPS Performance Analysis Report [93], the accuracy of a standard GPS device is within $3m$ with a 95% confidence, which can not be sufficient for some ADAS that require a more precise localization. Nevertheless, when a road has multiple lanes the problematic hardens. Indeed, the localization obtained from the Dead Reckoning technique is not enough precise to tell on what lane the vehicle is traveling. Hence, further information are used in order to obtain this knowledge, these information can be produced by visual sensors or by digital maps.

4.2/ Related Work

In order to localize itself, an autonomous vehicle is equipped with a set of sensors in order to perceive the surrounding world and to collect ambient information. Indeed, the underlying objective of a localization system in autonomous vehicles is to collect contextual information. To this end, the localization system requires an accurate digital representation of the surrounding world. In practice, localization of an autonomous vehicle can be done, by locating the vehicle with respect to some visual features such as lane markings, traffic signs. These visual landmarks can either be detected using on-board sensors, or be already stored in digital maps.
Concerning the Lane-Level Localization (LLL) the current literature is teeming with solutions that address this issue in a variety of manners. However, two techniques stand out. The first one depends solely on the detection of visual landmarks such as lane marking with an on-board sensor. For instance, lane marking systems are used in order to detect all the lanes on the road. Nevertheless, sometimes is not possible to detect all the lane markings on the road due to occlusions from other vehicles. Contrary to the first approach, the second approach depends on a more precise map. These maps store the accurate position of landmarks (e.g, lane marking). Therefore, the system have to match the detected landmarks with the one obtained with the visual information.

4.2.1/ Map aided approaches

To reduce sensor dependency, digital maps can store contextual information about the road. The amount of information stored depends on the scale of accuracy and detail displayed on the road network. Du *et al.*, [42] described three scales, commonly used for autonomous vehicle system, to cluster the existing maps as illustrated in Figure 4.2:

1. **Macroscale map** It represents the road network with a metric accuracy. These maps are used for route-planning problems and high guidance routines. Nevertheless, these maps provide the user meta information such as speed limitations or the number of lanes present on a given road. Besides, the road network is smoothed using clothoid curves which can give a general intuition on the shape of the road.

(a) (b) (c)

Figure 4.1: Types of maps generally used in intelligent transport system: (a) HERE map (© HERE), (b) OpenStreetMap (© (OpenStreetMap), (c) mesoscale map used in [115]. Details in text.

2. **Microscale map** It corresponds to the most accurate maps. These maps have centimetric accuracy representing the road network with dense information. Generally, Lidars are used to gather maximum information. Surprisingly, the fundamental benefit of these maps, which is their great information richness, is also their biggest disadvantage. Indeed, the density of information makes handling these maps difficult in order to isolate the point of interest.

3. **Mesoscale Mmap** McMaster *et al.*, [9] claimed that a map has to provide enough details about the environment without clutter up the user with unneeded information. Applied to previous definitions, microscale maps have to remove unneeded data, while Macroscale maps have to add more details to their database. This is where mesoscale maps stand. It is a trade-off between the two types of maps, i.e., it provides more accurate information about the road network, e.g., each lane is represented by a link. Meanwhile, it does not contain all the 3D representations of the road.

In the context of intelligent transport systems, mesoscale maps are the most used one (e.g., Lanelets map [85]) since they are the most suitable scale for intelligent vehicles, as it carries accurate information without being too dense. In addition, mesoscale maps have the merit of being easier to maintain compared to microscale maps. Therefore, a strong effort is currently made by the map's provider in order to meet these requirements. In order to have access to these contextual information, the autonomous car has to be located in regard to these maps. In view of the current state of affairs regarding the compactness of the maps and sensor availability, lane map-matching-based methods are widely used. Contrary to the research works regarding MM methods studied in Chapter 2, lane map-matching-based methods deal with a bigger amount of computation and location errors as the map becomes more complex. Indeed, for these techniques, ambiguous situations are more frequent. Take three-lane road, for example, and we want to assert the host lane position. In this case, if all the lane marking are stored in the map, we will have four links instead of one link representing the entire road. Therefore, in order to eliminate these ambiguities, several lane map-matching-based methods have been proposed.
In recent decades, several researchers have deeply investigated the idea of using cameras and HD maps to have a successful and precise localization algorithm [99, 139, 150]. Generally speaking, the vision-based map matching localization is a process that aligns

the perceived environment landmarks, such as lane lines with the stored landmarks in the map. In this context, Li *et al.*, [114, 115] presented a lane map-matching algorithm using a mesoscale with lane-level map accuracy. The lane MM method is based on MHT. Nevertheless, no external sensor has been used in order to perform the MM. The experimental results were conducted in terms of lane MM accuracy. Notwithstanding the significant results obtained, no number has been given in order to determine the position of the host lane. Within the same paradigm but in a different way, Kang *et al.* [163] proposed a lane map-based algorithm for lane-level localization using a mounted camera on an autonomous vehicle. The method relies on the detection lane marking the image frame. The detected lane marking was matched using the GPS trajectory with the map database that contains the center-line of each lane. The map matching method was based on a ICP-based rigid map. The results obtained in the experimental sections showed an improvement in the accuracy in terms of localization. Indeed, the average error obtained by the state-of-the-art device and that of the GPS was 2.340 m. This error was reduced to 0.475 m. Yet again, the value of the error obtained is not sufficient to clearly determine the correct lane on which the vehicle is traveling.

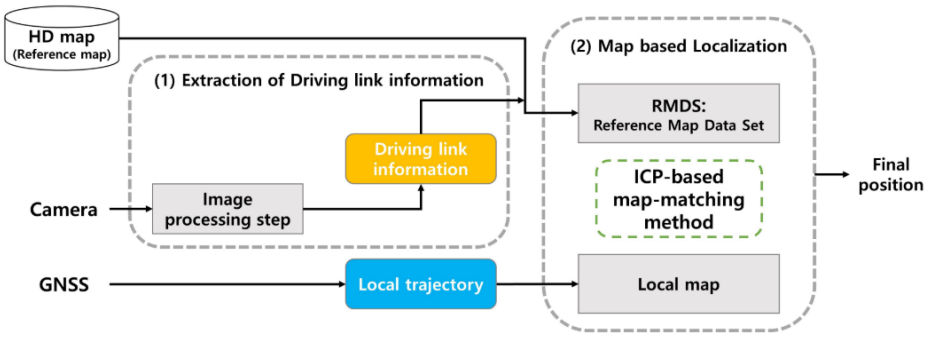


Figure 4.2: Framework proposed by Kang *et al.* [163]

In the same manner, Ghallabi *et al.* [127] presented a similar approach, in which a lane-level localization is performed using mesoscale map that is composed of many links (lane markings) in polylines form. The objective of the study is to match a polyline (lane marking) detected with an onboard sensor to the corresponding lane markings stored in the map. To perform such a task, a lidar sensor is used to detect the lane marking. Afterward, these detected lane markings are matched with the corresponding map. The map-marking strategy used relies on PF algorithm in which weights functions based on likelihood functions have been introduced. In terms of evaluation, several experiments have been conducted and results have been noted. To do so, a cross-track metric has been used to evaluate the accuracy of the matching strategies. Therefore, the results obtained represented the error between the matched lane marking in the map and the ground truth lane marking stored in the map. Finally, the authors come to the conclusion that the current results are promising and sufficient for Highway use-cases. However, the authors did not explicitly explain how the errors in lane marking position will affect the ego-vehicle localization. In addition, the errors that can be stored in the map are not taken into account which can also affect the accuracy obtained as pointed out by Welte *et al.*. [153].
Nevertheless, the majority of the lane-based map-matching techniques only consider the ego-lane marking lane in order to perform the map-matching. To overcome this issue,

Schur *et al.*, [105] propose to include all the lane markings in the lane-based-map-matching algorithms in which a PF is used. The performance of the general solution is considerably stable even in urban crowed scenarios. Indeed, the experiments conducted in South Korea showed that the average lateral error is about 0.50 m which is less than the width lane (3 m in Korea) which led the author to conclude that it is sufficient to recognize driving lanes. In the same spirit, Cui *et al.*, [95] propose to incorporate the lane-type information to the lane-based-map-matching algorithm with the type of lane marking. Nonetheless, these techniques claim a lane-level localization however, none of these techniques consider the ego-lane index (ELI) for map matching which is questionable if we take into consideration that in some cases, typically highway scenarios with multi-lanes, an ambiguity may exist in distinguishing the true lane. Indeed, all the lane marking shape are identical which results in increasing the difficulty for choosing the correct lane since there exists a strong ambiguity on choosing the right lane.
To address this issue, Lee *et al.*, [173] proposed an atypical approach. Indeed, the authors proposed a sequential framework, which is composed of different deep-learning blocks. The first DL block detects all the lane marking present in the image using DeepRoad Network. The second block is a LTSM network, the objective of this network is to identify the lane on which the vehicle is traveling. Finally, the lane information, which is the output of the latter DL block, is used in a lane-based map-matching, the full pipeline is clearly illustrated in Figure 4.3).

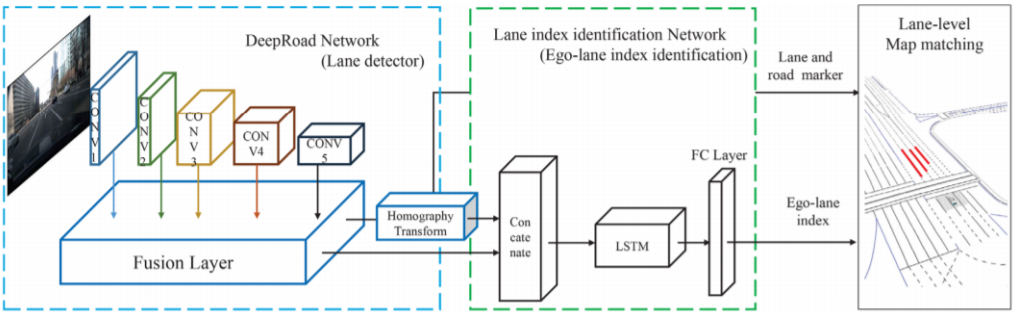


Figure 4.3: Framework proposed by lee *et al.*, [173]

4.2.2/ LANDMARKS APPROACHES

LLL have been the focus of numerous research over the past decades. In that regard, the current literature abounds with LLL techniques based on perception methods. In these approaches, relevant road level features are extracted from images, once these features extracted, they are fed into a high-level fusion framework.
In that context, Lu *et al.*, [91] estimate the probability of belonging to a lane, using lane change information and lane-markings detector. In the quest of LLL, Nieto *et al.*, [47] presented a LLL based from multiple-lanes detection. First, ego-lane detection is performed in order to detect the ego-lane marking and the lane geometry (curvature and lane width). Afterward, based on the estimation of the vanishing point, reconstruction of the geometry of the road is estimated, which gave an indication of the number of lanes. In addition, an assumption on the geometry of the road is made. This assumption formulates that the lanes on the same road share the same curvature and the same lane's width. Taking into account these considerations, adjacent lanes are hypothesized and tested, as illustrated

on Figure 4.4. The verification of these hypotheses is performed by a confidence level analysis, which is based on distance measurement.

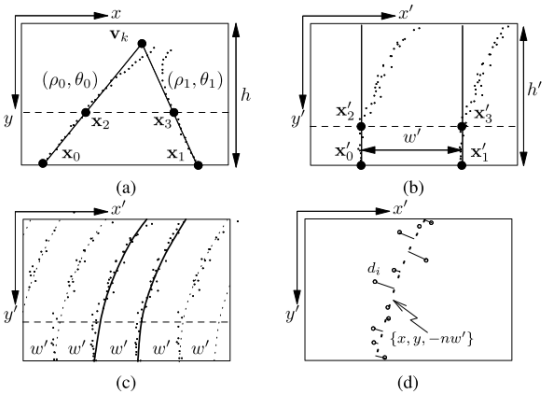


Figure 4.4: Framework proposed by Nieto *et al.*, [47]

In the quest of the LLL, Popescu *et al.*, [74] presented a probabilistic formulation of the problematic, first on intersection [69], then on more general road situations [74]. The lane marking information, together with some relevant visual landmarks like arrows in the lane, are fused as observation into a semi-fixed Bayesian Network, which is illustrated on Figure 4.5. The main objective of this network is to estimate instantaneous probabilities for each lane position hypothesis. The semi-fixed structure of this Bayesian Network allows the addition of more observations, e.g., adjacent lane marking and more arrows in the road. Eventually, the experimental results showed interesting results in identifying the ego-lane.

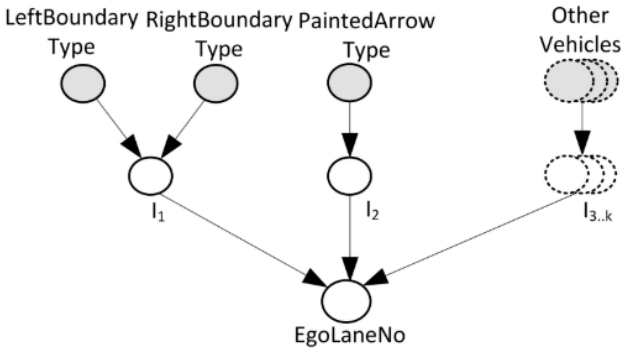


Figure 4.5: The semi-fixed Bayesian Network proposed by Popescu *et al.*, [74]

In the same manner, Ballardini *et al.*, [109] propose a LLL based on a Hidden Markov Model (HMM) to filter the outcome of a marking-lane detector based on stereo images. To do so, the HMM is modeled based on a lane detector, for which score functions are introduced depending on the reliability of the detector. Results on real datasets show very good results. Nevertheless, lane-changing situations have not been addressed. In addition, the probabilistic HMM calculation and formalization were not explicitly defined, leading to a non-intuitive definition for the emission and transition probabilities. In addition, one can notice that the dynamic aspect of the system, which includes the dynamics of the ego-vehicle, is not considered. Hence, no temporary relation between frames is taken into account. Furthermore, the major drawback of the previous technique appears when lane

marking information are not detected due to the presence of obstacles (e.g., vehicles). In that sense, the presented algorithms do not provide a recovery strategy.
Attempting to overcome the later difficulty. A more recent approach, presented by Volvo researchers Svensson *et al.*, [106], deals with the surrounding vehicle. The method is based on a Bayesian Filter that fuses the position of surrounding vehicles detected, a map that provides the lanes number, and an ego-lane marking that give the ego-lane geometry. The objective of the Bayesian filter is to infer the position of the ego-lane. The early tests show promising results. However, the main drawback of this technique is that it is tributary to the presence of vehicles. An illustration of this limitation is shown in Figure 4.6. As illustrated, the road is composed of four lanes, which means that four hypotheses exist for one ego-lane position. However, the detection of an adjacent vehicle (in red) leads to the elimination of one hypothesis. Indeed, if an adjacent vehicle is detected on the left then the ego-vehicle is not supposed to travel on the far left. Furthermore, real-world experimental results are missing to assess the efficiency of the approach especially when there is no surrounding vehicle.

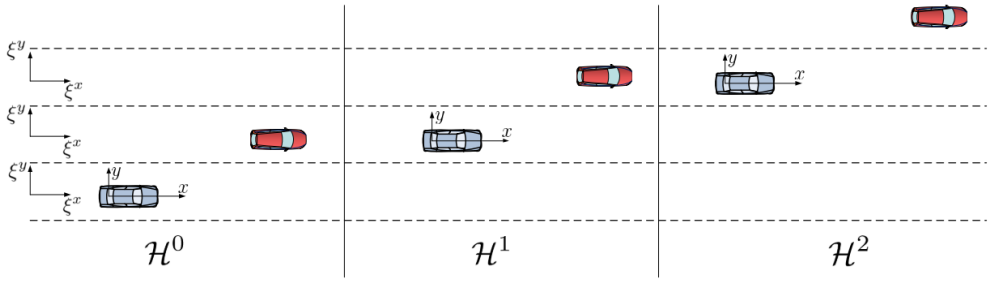


Figure 4.6: The LLL hypothesis of the ego-vehicle (in blue) and the adjacent vehicles detected (in red) (image credit [106]).

4.3/ CONCLUSION

In this chapter, literature about Lane-Level Localization has been studied and most relevant works have been presented. In the broadest sense, Lane-Level localization is a meaningful concept that can be related to two different problematics. The two paradigms lead to the same knowledge but differ in the methodology. The first is the knowledge of the lateral position of the autonomous vehicle with respect to the road. Solutions of this problem live in $\mathbb{R}$ and are usually solved using a map-aided approach. In this paradigm, lane-level map-matching algorithms are used to match the estimated position of an ego-vehicle, which can be estimated using Bayesian filter (e.g Kalman filter) with the proprioceptive sensors. This estimated position is then matched with a map. Generally speaking, the type of map used for this kind of task is the Mesoscale map [114, 115] using a lane-level map-matching algorithm. Contrary to the map-matching methods presented in Chapter 2, this kind of algorithm faces more difficult ambiguous cases. Typically for highway scenarios with multi-lanes, strong ambiguities do exist as all the lane marking shapes are identical. The second limitation of such a paradigm is in the type of map used. Indeed, these maps are relatively complex to build and cost-intensive, in addition to being difficult to use as they are not open-source.
The second paradigm uses a different methodology in order to solve the LLL problematic.

Indeed, the methods that belong to this group of paradigm articulate the knowledge of LLL as a classification problem. To do so, these methods rely on the relevant features that are present in the road scene, especially lane marking and adjacent vehicles. These relevant features are first detected and then fused in high-level fusion frameworks that are essentially based on a graphical probabilistic model, namely, Bayesian Network [69] or Hidden Markov Model (HMM) [109]. These probabilistic frameworks have the power and the ability to take into consideration uncertainties of the detected relevant features. Moreover, the physical constraints can be modeled in their network architecture making them real power tools to solve the problem. Contrary to the first paradigm, these methods rely solely on the exteroceptive sensors that are embedded in most of autonomous cars. Furthermore, they do not rely on the utilization of expensive maps. In the light of these reasons, we believe that the latter paradigm is more suited for the LLL especially for highway scenarios where the number of lanes can be greater than two. In addition, the utilization of the high-level fusion framework based on the graphical probabilistic method is modular to the addition of other sources of information, which enhanced the proposed output.
Now that all the parts of the ego-localization have been discussed in details, in the following chapter, we tackle the problematic of how these algorithms and methods have to be evaluated.

5

DATA-SETS FOR AUTONOMOUS VEHICLE

Contents

5.1 Introduction 69

5.2 Survey on datasets 70

5.3 How to evaluate a localization algorithm? 73

5.4 Conclusion 76

5.1/ INTRODUCTION

The global race to develop, evaluate, and deploy algorithms for autonomous vehicles has significantly heated up. In that sense, datasets have played a crucial role, by providing use-case scenarios with their respective ground truth. Indeed, in order to achieve reliability and robustness, autonomous vehicle systems have to be widely tested. To do that, the autonomous vehicle systems have to take into account all possible real-world scenarios and be relatively secure, necessitating extremely complex and diverse datasets. Furthermore, evaluating driving algorithms on real vehicles is often risky, expensive, and time-consuming. Therefore, the provision of training data that covers all possible real-world scenarios is crucial to autonomous drivings development and progress.
Thereby, due to their importance, datasets for autonomous vehicles have been widely studied [146, 161]. In the aforementioned section, and based on the literature, we present a broad spectrum of the different autonomous driving datasets which have been published up to date. We emphasize datasets that can be used for localization algorithms. Therefore, a focus will be given to datasets that are publicly accessible regardless of the type of sensors available. Indeed, the center of interest is the domain of application in terms of localization in the sense that was given in this thesis work, i.e., localization on a road (RLL), localization on a lane (LLL), and localization on the ego-lane (ELL). In the conclusion, a summary of the existing datasets is given specifying for each one their advantages and disadvantages.

5.2/ SURVEY ON DATASETS

The spread of autonomous vehicle projects leads to a multiplication of datasets. Each one is used in a range of applications for a specific block of an autonomous vehicle system. In the following, we will give a focus on datasets that can be used for a localization system. In order to identify these, we first define some components that are mostly shared between the datasets:

1. **The sensors used:** the type of sensors is highly dependent to the type of application test. Indeed, if the dataset contains GPS sensor, then the dataset can be used for RLL quantification.
2. **The environment:** typically, in this work we emphasize on road environment, and more precisely highway scenarios.
3. **The ground truth:** this component is crucial for providing a comparison between techniques.

In the context of the autonomous vehicle, KITTI is undoubtedly the most outstanding dataset and benchmark with the most comprehensive coverage of usage scenarios [146]. The KITTI vision benchmark was introduced by Geiger et al., [70]. The dataset was collected using a driving plate form equipped with several sensors: 2 grayscale and 2 color cameras, a Velodyne 3D laser, and high precision GPS/IMU (see Figure 5.1). The

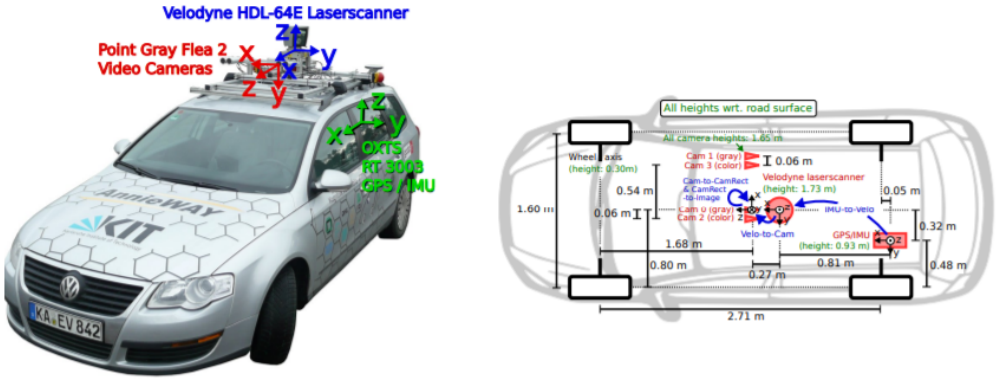


Figure 5.1: The recording vehicle used for the KITTI dataset (left). The mounting position of all the sensors (red) with the respect to the vehicle frame. Heights are noted in green and transformation in blue (image credit [70]).

datasets cover different road scenarios, i.e., urban, highway. In addition to several traffic densities and several lane configurations, i.e., urban marked, urban unmarked. The KITTI provides annotated data and therefore has been used to benchmark several disciplines in the localization, i.e., slam localization, lane detection. The major drawback of the KITTI is that it is limited in size. Which means, that the datasets can not be utilized for localization problematic like Lane-Level Localization (LLL). Indeed, the lanes number is not sufficiently high in order to be relevant to the evaluation. In addition, the lane change scenario is missing. Therefore, the KITTI dataset is usually used mostly for evaluation and fine-tuning.
Concerning the lane detection datasets, several projects have launched on that subject.

However, there is no real consensus on how lane detection datasets can be presented. Indeed, it can be a special case of object detection or in semantic segmentation, depending on the type of annotation type used. While the majority of datasets use spline lines to describe lanes, other datasets use pixel-wise annotation. Culane is a lane datasets released by the University of Hong-Kong for the academic world. The datasets were recorded in the city of Beijing on an acquisition vehicle that had 6 different cameras mounted. In total, 133,235 frames have been recorded in various road scenarios including urban and highway roads. However, no other sensor data, like lidar, GPS, is available. Lane markings are either interpreted with cubic splines, or they are occluded by vehicles, the lanes are also interpreted depending on the context as shown in Figure 5.2.

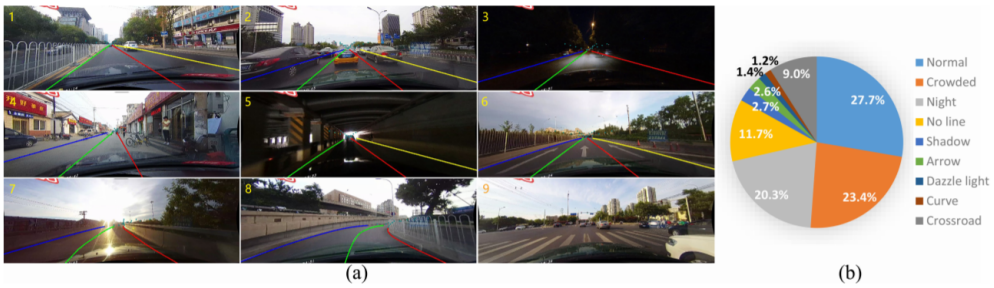
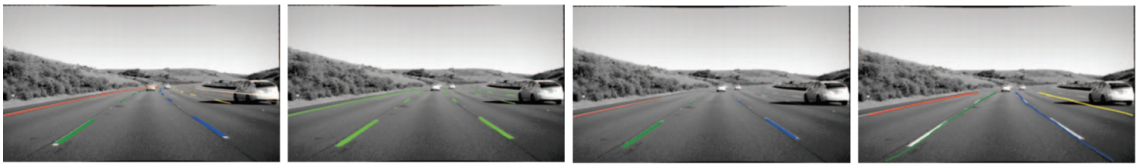


Figure 5.2: (a) different scenarios for the Culane dataset, (b) proportion of the different scenarios (image credit [138]

The Universidade Federal do Espırito Santo in Brazil launched the Ego-Lane Analysis System (ELAS) [110] dataset. It was generated using camera images from over 20 different scenes (over 15,000 frames) and a multitude of scenarios (urban route, highways, traffic, shadows, and so on). The dataset was manually labeled and made publicly accessible to encourage the scientific community to evaluate a number of algorithms related to the AVs (i.e., lane estimation; road markings; lane marking type (LMT); crosswalks and adjacent lanes; etc.) [110].
In the same spirit, the California Institute of Technology launched the Caltech Lanes dataset [40]. The dataset is composed of 1.2k frames decomposed of different clips. The frames are recorded using a camera mounted on the vehicle. The datasets do contain annotation about spline lane marking. However, other than a camera, the dataset does not provide more sensor data. Another dataset for lane detection is the TuSimple dataset. The dataset was released for the lane detection challenge that was held in CVPR'17. The data is collected using a camera in medium and good weather conditions and on different days time in a highway road that has multiple lanes. Even though the database contains 7k video clips, the length of every video is $1s$, which can make the validation of a LLL algorithm complicated. Regarding of the ground truth, polyline annotation is used for lane marking.
With the objective of proving one of the largest high-quality lane marker datasets, Bosch launched the Unsupervised LLAMAS dataset [141] which consists of 350 km recorded drives images in purely highway scenarios. In addition to the length of the dataset, the originality of the proposal lies in the label annotation. Contrary to the existing works, the annotations were auto-generated (as illustrated in Figure 5.3). Furthermore, the annotation ranges from the pixel-level annotations of dashed lane markers to the $2D$ and $3D$ endpoints for each marker. The major weakness of this dataset is the absence of other

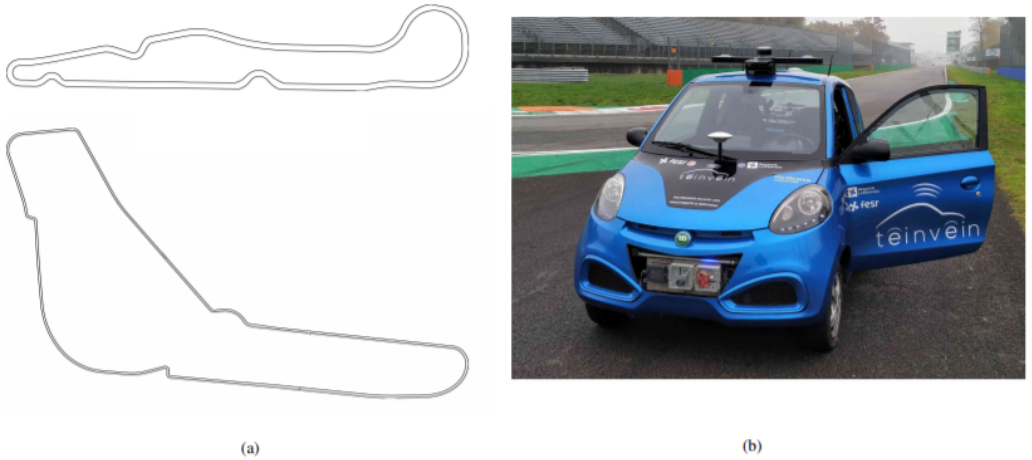
sensors information such as Lidar or GNSS data.



(a) Auto-generated labels (b) Binary segmentation (c) Multi-class segmentation (d) Spline regression

Figure 5.3: (a) represents the labeled lane image produced by the automated labeling framework. (b-d) shows three representation for the lane marking. (image credit [141]

More recently, Liang et al., [167] presented the TTLane Dataset with more than 10k images. The main contributions lie in the comprehensive annotation for the type of marking. Indeed, the TTlane dataset offers a more concise type of marking that covers white solid, white dash, yellow solid, yellow dash, and double lines. These datasets are highly used for lane detection challenges.
Moreover, if we consider the lane detection problematic as a sub-task of the ego-vehicle localization, these datasets do not report a way to obtain a ground truth measure of the relative pose of the vehicle within the lane. Nevertheless, recently, a study presented by Cuadrano et al., [158] expresses these concerns. Indeed, the authors presented an ego-lane localization based on ego-lane marking detection, in which they presented a dataset that contained ground truth about the position of each line marking in the scene. Naturally, this task is hardly possible on public roads. Therefore, the authors performed the experiments on circuit tracks as illustrated in Figure 5.4. The circuit is a simplified environment of a highway scenario. Indeed, no other vehicle is recorded during the driving test. However, the database can be used in order to evaluate an ELL algorithm.



(a) (b)

Figure 5.4: (a) The two driving test circuits (b) The vehicle used for the experiment (Image credit [158]

In the same context, recently, major companies have started the race for autonomous vehicles and therefore have started making their own annotated datasets. Baidu launched the autonomous driving project called "Apollo" in which the AppolloScape dataset was released [129]. The acquisition platform system is composed of two VUX-1HA laser scanners, VMX-CS6 camera system (two front cameras are used with resolution 3384 x

2710 pixels), and IMU/GNSS . The whole system has been mounted on the top of a SUV as shown in Figure 5.5. The AppolloScape dataset consists of annotated street view images for semantic (144k images) and instance segmentation (90k images), lane detection (160k images), car detection (70k), and tracking of traffic participants (100k images). The dataset allows evaluating the performance of methods in various weather conditions and at different day times (Figure 5.5). In the same spirit, the company Nutonomy released the NuScenes dataset [156], which consists of over 1 million camera images. Nevertheless, both AppolloScape and NuScenes have been recorded in one city limiting the diversity. Finally, these two datasets are more oriented for vehicle detection than for localization using lane marking.

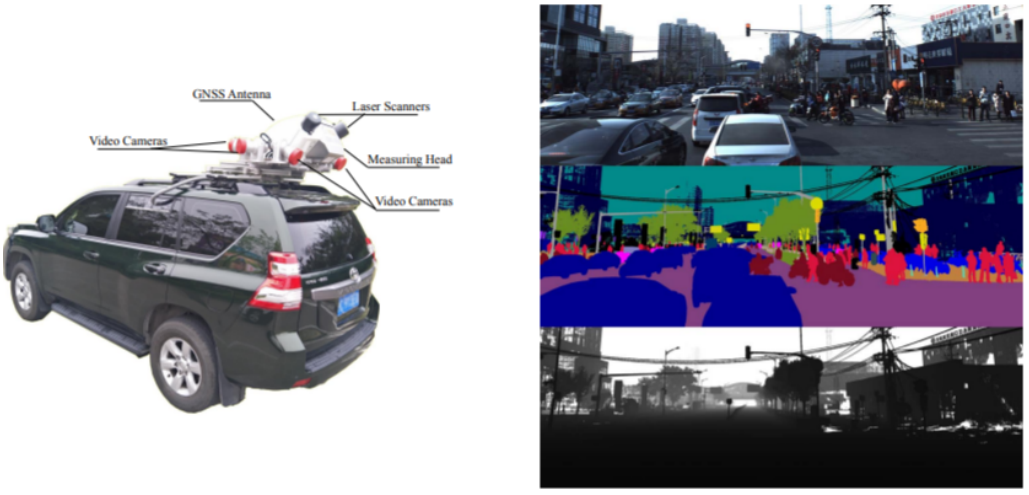


Figure 5.5: The recording vehicle used for the AppoloScape dataset (left). In the right, an example of color image (top), 2D semantic label (middle), and depth map for the static background (bottom) (image credit [129]).

The list of the datasets that have been published is vast, and by the time this manuscript we will be finished, other datasets will be published. Therefore, in the Table 5.1 we summarized the existing datasets that are relevant to our work, this list is non-exhaustive. However, the cited datasets are the one which allows access to their data. Therefore, the classification is made depending on the sensors available (video camera, lidar, GNSS), the scenarios covered (urban, highway, and rural), and the ground truth annotation available (pixel-wise, spline annotation).

5.3/ HOW TO EVALUATE A LOCALIZATION ALGORITHM?

Leafing through the literature, it appears that there exists no real consensus on how a localization algorithm should be evaluated. Indeed, depending on the application one evaluation metric can be more relevant to another. Nevertheless, sometimes the most used metrics are not fitted for all the applications. Therefore, in the following, we will discuss in detail on what are the features that have to be evaluated in a localization algorithm, and what are the most relevant metrics. Before heading further, we start by introducing the most known metric used for a localization system which is composed of

Table 5.1: Data-sets for localization system. Only the relevant datasets related to our work are presented. The abbreviation used are the following: Vi: Video, Li: LiDAR, U: Urban, H: Highway, Px: Pixel

Name	**Sensors**			**Scenarios**			**Annotations**	
	Vi	Li	GPS	U	H	Ru	Px	Sp
CalTech Lanes (2008) [40]	✓			✓	✓			✓
Honda Road Marking (2012) [75]	✓			✓	✓			
KITTI Road (2013) [77]	✓	✓	✓	✓	✓	✓	✓	
VPGNet (2017) [113]	✓			✓	✓		✓	
TuSimple (2017) [122]	✓				✓	✓		✓
ELAS (2017) [110]	✓			✓	✓	✓	✓	
A4 Italy (2017) [109]	✓		✓		✓			
VPGNet (2017) [113]	✓		✓	✓	✓			✓
TRoM (2017) [116]	✓			✓	✓			✓
ApolloScape (2018) [129]	✓	✓	✓	✓	✓	✓	✓	
CULane (2018) [138]	✓			✓	✓	✓		✓
BDD (2018) [140]	✓	✓	✓	✓	✓	✓	✓	
Unsupervised Llamas(2019) [141]	✓				✓		✓	✓
TTLane dataset (2020) [167]	✓			✓	✓		✓	✓
NuScenes (2020) [156]	✓	✓	✓	✓	✓			

three components: RLL, ELL, and LLL.
In computer science, several metrics have been introduced in order to evaluate algorithms. For example, a classification algorithm is trained to perform digit recognition (for example the number 5) and we want to evaluate its performance. The general idea is to count the number of times the classifier correctly classified a number as five, which is called True Positive (TP). But also, the number of times it classified a non-five number as a non-five-number, which is called True Negative (TN). In addition to that, we have to count the number of times a non-five number was classified as five (False Positive (FP)), and the number of times a five number was classified as non-five (False Negative (FN)). A perfect classifier would have only TP and TN. However, depending on the application these scores are not sufficient to give an evaluation of an algorithm. Therefore, more sophisticated metrics have been introduced, mainly, four quantitative performance metrics, i.e., Precision, Recall, F1 Score, Accuracy. Naturally, a combination of these metrics can be found in the literature to obtain new metrics more relevant depending on the applica-

tion [124].

Definition. (Precision) is the fraction of TP out of the sum of TP and False Positive (FP):

$$Precision = \frac{TP}{TP + FP}$$

Definition. (Recall) or also known as true positive rate, is the fraction of TP out of the sum of TP and False Negative (FN):

$$Recall = \frac{TP}{TP + FN}$$

Definition. (F1 Score) is the harmonic mean of the recall and the precision:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Definition. (Accuracy) is the fraction between the correct classification TP and True Negative (TN) out of the sum of all classification:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

The figure 5.6 helps to understand the meaning of these metrics.

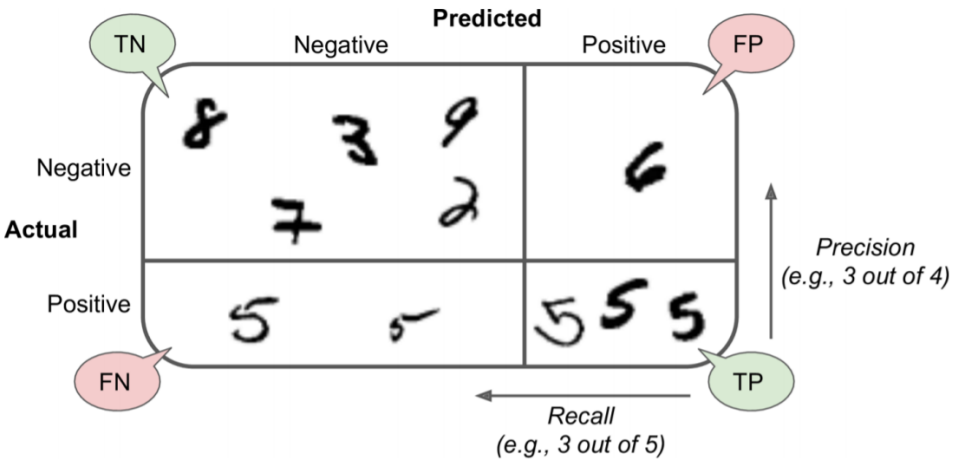


Figure 5.6: A graphical illustration to help understand the previous example.(image credit [143]

Going back to the localization system, these metrics are also used in most benchmarks that used camera images, i.e., KITTI, Tusimple. However, concerning the detection of lane marking these metrics are not well suited. Indeed, these metrics are based on pixel-wise annotation, which means, each pixel is classified: either it is a lane marking or not. Therefore, the lane detection algorithm has to outputs pixel wises annotation of the image, which is not always the case. Considering the work presented in [18], the pixel-level annotation is not suitable. Indeed, the authors have a higher-level representation of the lane, as the lane is represented as a polynomial. In addition to that, these metrics

do not take into account the shape and the distance between the detected lane and the ground truth.
Nevertheless, when dealing with 3D data (e.g., lidar data), the metrics used are different. Indeed, it is more interesting to use metric errors based on the difference between the estimated lane marking and the current lane marking, which are more relevant to the field of the localization system. In that context, several metrics have been studied, namely, mean squared error (MSE), root-mean-square error (RMSE):

Definition. (MSE) is used to measure the difference between values obtained by a predicted model and the ground truth values. Mathematically speaking, the RMSE represents the quadratic mean of the difference.
If a vector Y_e is the estimated vector of n components, and Y_g the corresponding ground truth, then the MSE is computed as following:

$$MSE = \frac{1}{n}\sum_{1}^{n}(Y_e - Y_g)^2$$

Definition. (RMSE) is also used to measure the difference between values obtained by a predicted model and the ground truth values. The RMSE the root of the MSE.

$$RMSE = \sqrt{MSE}$$

These two metrics, and their derivatives, have been widely used for RLL problematics. In these topics, the ground truth is obtained from a more accurate solution (GPS RTK + more accurate IMU) and compared to the RLL solution. Considering, the ELL, the dataset has to contain the ground truth representation in the world coordinate. To the best of the author's knowledge, there is no dataset that contains such representation.

5.4/ CONCLUSION

In recent years, the heated competition to deploy self-driving cars has taken a major turn. The community that embraces researchers, major car manufacturers, as well as giant tech companies, have started to develop functional algorithm blocks for self-driving cars. They share the same goal; which is to deploy a completely self-driving car on a public road. Nevertheless, before getting to this end, they must all test and evaluate their solutions. As a consequence, the need for a solution to systematically evaluate their algorithms without deploying them on real vehicles has been a critical subject in the field of the autonomous vehicle. A solution that has been widely adopted is to test and evaluate autonomous algorithms on available public datasets, by doing so researchers can evaluate their algorithms on real-world scenarios without taking the risk of deploying their solutions on real cars. In addition to that, researchers can evaluate and compare their techniques to their colleagues, which creates a constructive and healthy competition among them in order to finally converge to the best solution to be deployed on the autonomous car.
In this chapter, literature about the datasets available that can serve to evaluate and localization systems has been presented. To realize such a review, we classified the datasets depending on several criteria, namely, sensors data available (i.e., Lidar, camera images, GNSS receiver, IMU), the scenarios covered (mainly highway, urban, and rural), the type of annotation used (pixel-wise, spline). Depending on the application and the type of

Table 5.2: Requirements needed to evaluate each part of the localization system (RLL, ELL, and LLL), with the corresponding datasets available with respect to the Table 5.1. The abbreviation used are the following: U: Urban, H: Highway, R: Rural

	Sensors	**Scenarios**	**Annotations and Metrics**	**Datasets that can be used**
RLL	GPS+IMU	H, U, R	MSE, RMSE and derivatives	KITTI Road (2013) [77]
				ApolloScape (2018) [129]
				A4 Italy (2017) [109]
ELL	Camera or Lidar + IMU	H, U, R	MSE, RMSE and derivatives	KITTI Road (2013) [77]
				Unsupervised Llamas(2019) [141]
				ApolloScape (2018) [129]
				ELAS (2017) [110]
				VPGNet (2017) [113]
				A4 Italy (2017) [109]
				TuSimple (2017) [122]
				NuScenes (2020) [156]
LLL	Camera or Lidar + IMU	H	F1 score, Accuracy, Precision, Recall and derivatives	KITTI Road (2013) [77]
				Unsupervised Llamas(2019) [141]
				ELAS (2017) [110]
				A4 Italy (2017) [109]

annotation used, specific metrics are used. Nevertheless, the most know benchmarks, KITTI, TuSimple, CuLane, AppoloScape, are dominated by one paradigm of evaluation. Indeed, for these datasets lane markings are annotated at a pixel level. As consequence and as pointed out by Cuadrano *et al.*. [158] none of these datasets contains a world representation that can be relevant for AVs control systems. In addition to that, presenting a dataset that contains this type of annotation and representation is hardly possible unless privatization of the public road is done.

Concerning the localization components (RLL ELL, and LLL), we summarize the needs for each component in terms of sensors data available, scenarios covered, and annotation and metric used. The summary of these requirements is noted in Table 5.2. One can note that this Table takes into consideration the same datasets presented in Table 5.1.

In conclusion, there is no real consensus on how a localization algorithm has to be evaluated. In our work, we will first evaluate each part of the localization solution independently. Afterward, an evaluation of the whole system is given, due to the sequential nature of our proposed architecture, the output of our algorithm is based on large datasets that have been collected on the Clermont-Ferrand region, the latter will be discussed in Chapter 9.

CONCLUSION

In this part, state-of-the-art review on the different localization system have been proposed. In chapter 2, review of the localization techniques related to the Road Level Localization (RLL) have been discussed. Mainly, these techniques rely on MM algorithm. The objective of these MM algorithms is to match the vehicle localization, which estimated using a GNSS receiver and internal information about the vehicle, with a digital map. To attain these objectives, diverse approaches have been studied. The most know methods relies on probabilistic framework, unquestionably the probabilistic reasoning is suitable for this kind of problematic. Indeed, it allows taking into consideration the uncertainties that are inherent to GNSS receiver in order to correctly choose the right road on which the vehicle is traveling.
In the chapter 3, we discussed methods that are used in order to locate the ego-vehicle in its own lane. The state-of-the-art review conducted showed that this localization knowledge is achieved by detection of the ego-lane marking. In order to detect these visual landmarks, two families of approaches exist. The first one, is the "classical" approach, or the model-driven approach. The methods that fall into this category share the same architecture. In this architecture, the lane-marking detection is achieved in sequential fashion via several processing blocks. Each block is responsible for one unique task (preprocessing, fitting procedure...etc). By decomposing the problematic into sub-task, this first category is more modular. However, it suffers from its strength, as some processing block (for example the fitting procedure) are not generalized to all type of road. In the other hand, the second category called the data-driven approach, is a monolithic method in which the ego-lane marking is performed in one stroke. Even if these methods show great results, as they lead the majority of benchmarks. These methods are presented like black boxes in which the processing steps are "hidden". Furthermore, these methods rely on DL techniques and hence suffer from their limitations: the availability of the training data, the generalization problematic.
The Chapter 4 discussed the last component of the localization algorithm which is the Lane-Level Localization (LLL). To perform such a task, the majority of the works presented follows two distinct paradigms. The first relies on MM methods on more accurate map (which are commonly denoted as HD maps). The problematic is the same as the one studied in Chapter 2. However, the richness of the map increases the degree of complexity of the MM. Indeed, the ambiguous cases are more recurrent compared to the low quality map. In addition to that, these methods are dependent on the map and its availability, which is not always the case considering the high cost of building and updating these type of maps. The second paradigm emphasized on the available landmarks on the road in order to infer the position of the ego-lane. Mainly, lane marking and adjacent vehicle are detected. These detected landmarks are then fused in a fusion framework (for example BN) in order to determine the LLL.
Finally, the last chapter 5 was dedicated to the metrics and the datasets that can and should be used in order to evaluate a localization algorithm. Due to the heterogeneous nature of each block compositing the localization algorithm, several metrics have to be used. Concerning the datasets, a list of available dataset have been presented in Table 5.1. The objective of this table was to answer the following question: what are the datasets that can be used in order to evaluate a localization algorithm ? Therefore, to answer to these questions, several parameters and factors have to be taken into account, namely, the scenarios studied (highway, Urban), the data available (GNSS data, lidar, camera) and the availability of the ground truth. As a consequence, there is no dataset

that satisfy all these requirements. To this end, we will propose our own collected dataset in Chapter 9.

II

End-to-End Probabilistic Ego-Vehicle Localization Framework

OVERVIEW

The state-of-the-art review on localization systems presented in the part 1 led to the conclusion that in order to achieve a reliable ego-localization, a localization system has to satisfy the knowledge of three localization components, namely, Road Level Localization (RLL), Ego-Lane Level Localization (ELL), and Lane-Level Localization (LLL). As a result, the contributions to the thesis can be categorized into three parts: the global functional architecture of the developed system is shown in Figure 5.7.
Chapter 6 details the Road Level Localization (RLL) solution presented, Chapter 7 details the Ego-Lane Level Localization (ELL) algorithm, and Chapter 8 details the Lane-Level Localization (LLL) framework presented. Throughout the chapters, demonstrative examples are shown to validate each part. In order that this dissertation becomes at maximum self-contained, the main assumption and background work will be given in this section.

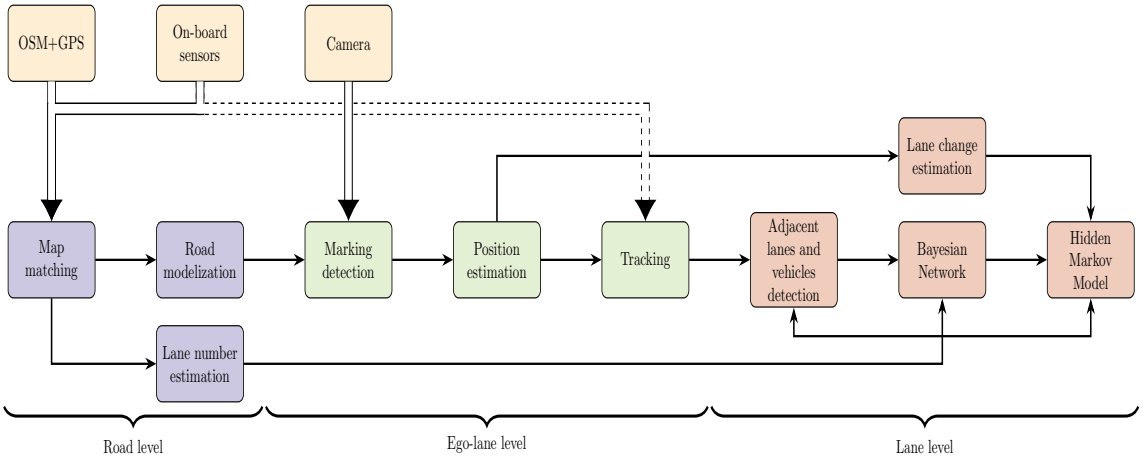


Figure 5.7: End-To-End Probabilistic Framework For Ego-Localization proposed in this work. Each part of the framework are the focus and the main contributions of the Ph.D. thesis.

6

PROBABILISTIC MAP MATCHING ALGORITHM (PMMA)

Contents

6.1 Problem Formulation 83
6.2 OpenStreetMap Dataset 84
6.2.1 Nodes 85
6.2.2 Ways 85
6.2.3 Relations 85
6.3 Proposed solution 86
6.3.1 Pre-Processing 86
6.3.2 Discrimination 88
6.3.3 Selection Stage 90
6.3.4 Functioning of the algorithm 93
6.3.5 The enhanced Probabilistic Map Matching Algorithm (PMMA) . . 95
6.4 Results and discussion 98
6.5 Conclusion 102

6.1/ PROBLEM FORMULATION

A reliable Map-Matching (MM) algorithm for any localization system, according to [39, 157], requires the ability to solve any road configuration (highway with insertions, exits...) by taking into consideration the inherent uncertainties resulting from the sensors and the map. Furthermore, it has to deal with the ambiguous situation caused by unexpected situations while finding the right balance between accuracy and computational expenses. The state-of-the-art review on Road Level Localization (RLL) methods, and more precisely on Map-Matching (MM) methods led to the conclusion that hybrid architecture that includes both the deterministic method and the probabilistic method comes out to be the best solution regarding the defined characteristics. Indeed, it has the ability to deal with the limitations raised from both methods. Therefore, we present a hybrid architecture as a response to the defined characteristics (Cf. Section 6.3.5).

The initial map-matching process is performed using the deterministic method. The objective of this process is to select road segments that fall within a circle-shaped selection area based on the errors associated with the navigation device, the quality of the map and the configuration of the vehicle pose. In addition to that, as pointed out by Quddus et al., [39], the algorithm has to take into consideration road design parameters, such as turn restrictions, roadway classification (such as one-way, two-way), and the number of lanes in the MM process. Consequently, our approach takes into consideration these attributes' data depending on their availability.
Once the initial map-matching process, the incompatible road segments are eliminated. For some cases, this initial process step is sufficient to eliminate all the incompatible road segments. However, in some ambiguous cases, the number of remaining road segments is greater than one. Therefore, a selection strategy has to be adopted. As discussed in Chapter 2, the selection strategy has to take into account inaccurate data sensors information in addition to both geometric and topological errors in the map used. In that regard, probabilistic approaches come out as the best solution for their well-known ability to account for uncertainties to select the best road segment. Hence, in this work, we rely on the use of Hidden Markov Model (HMM) in order to model the road network ambiguities. The trade-off between accuracy and time complexity offered by this type of network is more relevant to our scope of utilization. Contrary to other works, the modeling of our HMM is intuitive and comes out directly from the theory. More important, we do not rely on the Viterbi algorithm to solve the MM problematic as the use of this algorithm is time-consuming.
The chapter is structured as follows: Section 6.2 introduces the data model of the OSM geodata. Section 6.3 describes the Map matching algorithm used. Section 6.4 presents the real-world experimental results.

6.2/ OpenStreetMap Dataset

The unavoidable element to perform the MM is the map. By definition a map is a directed graph $G = (V, E)$, in which a vertex $v = (x, y) \in V$ represents an intersection or a road end, and an edge $e = (s, e, l) \in E$ is a directed road starting from vertices s to e with a polyline l represented by a sequence of connected segments[157]. The latter definition stands for the majority of the map presented in the literature. For our work, we solely utilize the OpenSteetMap (OSM). Therefore, for the sake of clarity, a brief description of this map is given. The elements that composed this map as well as the main features that have been utilized are presented.
OpenStreetMap (http://www.openstreetmap.org/) is a collaborative project that aims to create a free global geographic database. Its ultimate objective is to compile a database that contains all the possible geographic features of the earth. While the project began by recording roads and streets it has since expanded to include houses, rivers, pipelines, woods, beaches, postboxes, and even individual trees [62].
The default format for representing the data model is XML, and it can be only downloaded data from the OpenStreetMap server in that format. For local sections of the planet, a XML file containing the latest revision of the OSM map can be downloaded from the official website of OpenStreetMap. Furthermore, regular updates of the geodata are available on that website.
The data model is made up of three basic primitives. Each of them is tagged to contain key-value pairs, the geographical features they represent can be identified. The OSM

data model follows the definition of a classical map. Indeed, taking the mathematical definition, the OSM data model is a directed graph, it consists of vertices and edges. Depending on what features in the real world they're modeling, these parts of the graph can be related or separated.
Concerning the three basic primitives that compose the whole OSM database, there are defined in Nodes, Ways, and Relations.

6.2.1/ NODES

Nodes n_i are point-shaped geometric elements which are used to represent space points in terms of latitude (lat) and longitude (lon). Moreover, Nodes are the only primitives with position information. Therefore, all other primitives depend on Nodes to locate themselves; for example, Nodes are the basic points to represent the geometry of the Way.

6.2.2/ WAYS

Ways are a list of Nodes that are arranged in a certain order. A Way has to have at least 2 Nodes. The distribution of the Nodes is not uniform in the Way. Hence the distance between two consecutive Nodes is variable. Ways are used to model line-shaped geometric objects like roads, railways, pedestrian roads. Throughout this work, a single Way is defined as follows:

$$W = (id_w, N_w, T_w) \tag{6.1}$$

Where id_w denotes a unique identification number of the Way. In addition, the set N_w regroup the m Nodes n_i representing the geometry of the Way as follows:

$$N_w = \{n_1, n_2, ..., n_m\} \tag{6.2}$$

To specify the semantic of each Way, a subset T_w of n tags is related to the Way. The tags contain the meta-information about the nature of the way, i.e, the lanes number, the type of road, the speed limitation. According to the OSM specifications, each Way can have up to 255 tags. These tags are defined as follows:

$$T_n = \{t_1, t_2, ..., t_n\} \tag{6.3}$$

where t_i indicates a single tag. Each tag consists of two elements, a key k and the corresponding value v :

$$t_i = (k, v) \tag{6.4}$$

As illustrated in Figure 6.1 an example of an OSM Way with its corresponding tags.

6.2.3/ RELATIONS

The element relation describes the relationship between Nodes as well as Ways. This element is not used in this thesis. For more information, we invite the reader to refer to [43].
Now the OSM database model has been presented, the following section introduced our proposed MM solution algorithm.

Key k	Value v
highway	motorway
lanes	4
oneway	yes
ref	A5

Figure 6.1: Example of an OSM Way with its corresponding tags, keys and values.

Table 6.1: Used OSM tags

Description	Key	Value
road speed limit in km/h	maxspeed	number
total number of physical lanes of the Way	lanes	number

6.3/ PROPOSED SOLUTION

The overall MM architecture proposed is illustrated in the figure 6.2. The component blocks structuring the proposed algorithm are the following:

- **Inputs:** two main sources of information are used, namely, the GPS data coming from a classical, and hence inaccurate GNSS receiver, and the OSM geodata, which are retrieved from the OSM dataset.
- **Processing blocks:** composed of three parts, the preprocessing part, the discrimination stage, and the selection stage. These three combined blocks are the core of the PMMA.
- **Outputs:** the algorithm outputs the Way on which the vehicle is traveling in addition to all the meta-information that is stored in the tags composing this Way.

In the following section a description of each component block is given.

6.3.1/ PRE-PROCESSING

As presented earlier the OSM database is composed of mainly three elements: nodes, ways, and relations. The ways are a set of order nodes that can represent geometric features like roads, but also other elements like railways and pedestrian roads. This abundance of data is the strength of OSM. However, if we consider the Ways in which the vehicle can not travel, i.e., railways, pedestrian roads, the MM task would be complex, and the outputs will be jeopardized by the multitude of choice for the Ways. Therefore, the objective of the pre-processing stage is to reduce the complexity of the MM procedure by

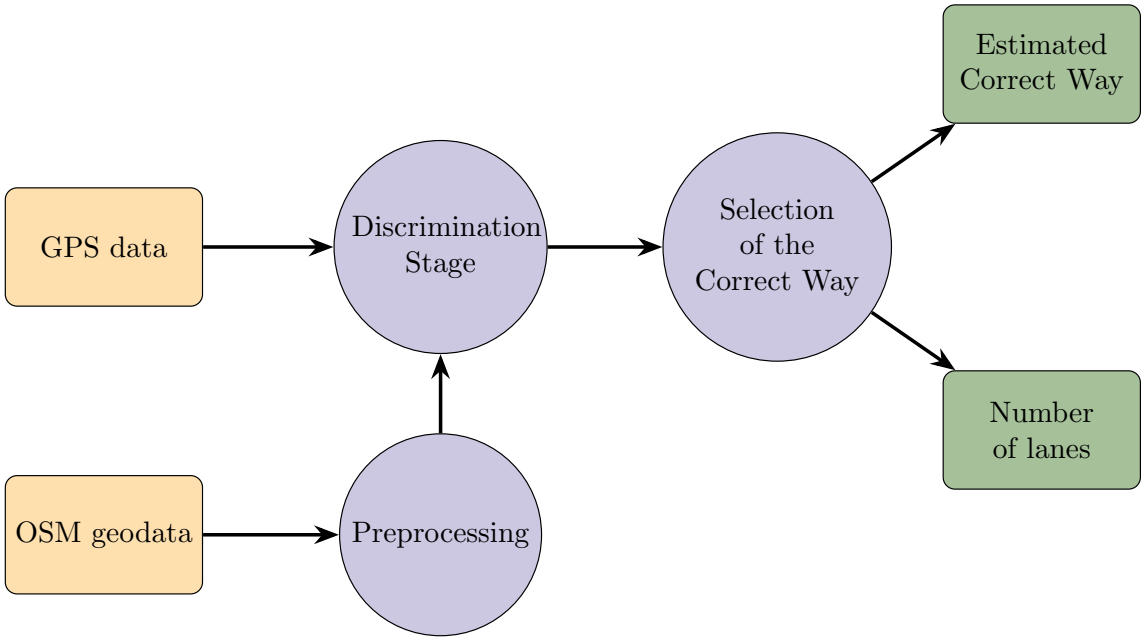
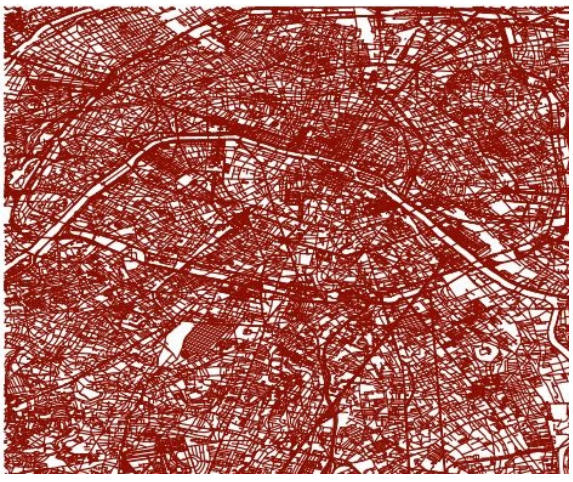
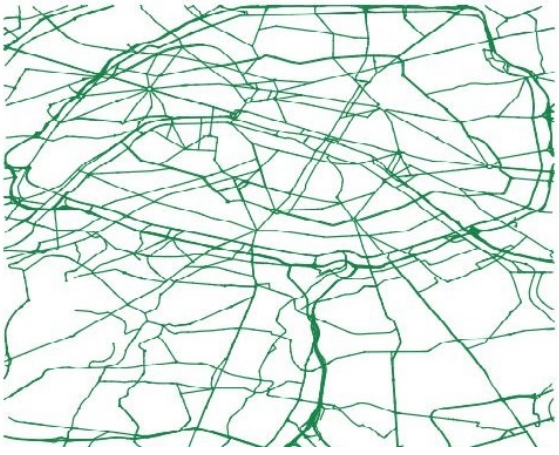


Figure 6.2: Flowchart of the proposed PMMA [164]. The inputs are highlighted in yellow, the outputs in green, and the processing stages are in blue.

reducing the size of the OSM database by eliminating the Ways on which the vehicle can not travel. To do so, we used the set of tags provided by the OSM database in order to eliminated ways that do not represent roads. The list of the used tags listed in Table 6.1. An example of an output of a discrimination stage is illustrated in Figure 6.3, as shown the density of ways in Figure 6.3(a) is highly reduced as shown in Figure 6.3(b)



(a) Before preprocessing



(b) After preprocessing

Figure 6.3: OSM map before preprocessing stage Figure 6.3a and after preprocessing stage (elimination of the ways that does not represent roads on which the ego-vehicle can travel) Figure 6.3b

6.3.2/ DISCRIMINATION

Now the pre-processing step is performed, the output map is composed of a set of Ways, which are composed of a list of nodes. Each Way can either be composed of two nodes, or more nodes. Therefore, the geometric representation of the Ways is not homogeneous. In order to have a homogeneous representation of the Ways, in our work, we represent each Way by a set of segments S_w with each Way composed at least of one segment. As a consequence, the Equation 6.1 is then rewritten as follows:

$$W = (id_w, S_w, T_w) \tag{6.5}$$

$$S_w = \{s_1, s_2, ..., s_m\} \tag{6.6}$$

$$s_i = (n_{i1}, n_{i2})^T \tag{6.7}$$

s_i being each segment composing the Ways. One can note that two consecutive segments (e.g., s_1 and s_2 share one node). This modeling allows a homogeneous representation of the map and simplifies the MM procedure. Indeed, to perform the MM it is necessary to select the right segment. In other words, belonging to a segment is equivalent to belonging to an OSM Way. So the map matching task can be reformulated as matching an estimated vehicle position with a segment. In the remaining sections, words segment and way are switchable.
After this modeling is completed, the map now is made up of a list of segments that have the characteristics of Ways (in terms of tags) but have a homogeneous representation. Accordingly, the objective of the discrimination stage is to remove all the ways which are not suitable for map matching depending on several factors:

- distance to road,
- angle difference,
- speed difference

6.3.2.1/ DISTANCE TO ROAD

The smallest distance d_{min} between a geographical object in 2D space p and a segment s, with $s = (n_1, n_2)$, $n_1 = (x_1, y_1)^T$, $n_2 = (x_2, y_2)^T$, and $p = (x_p, y_p)^T$ is illustrated in Figure 6.4.

In order to calculate this distance d_{min}, we look for the point belonging to the segment s which is closest to the point p, this point is denoted p'.
To begin, we compute p' on the line (s) parametrized by the real number t such that $p' = n_1 + t.s$ and:

$$t = \frac{\overrightarrow{n_1 p} \cdot \overrightarrow{n_1 n_2}}{|n_1 p||n_1 n_2|} \tag{6.8}$$

"$\cdot$" being the dot product of two vectors. We ascertain that the point p' belongs to the segment s, i.e. that means $t \in [0; 1]$. If p' does not belong to the segment, we bring it back to the nearest end:

$$t = min(max(0, t), 1) \tag{6.9}$$

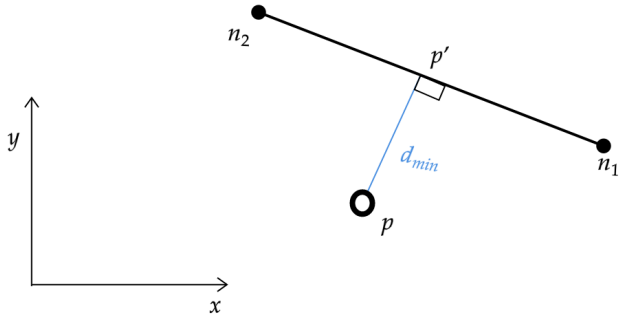


Figure 6.4: Closest distance between a geographical object p and a segment $s = (n_1, n_2)$.

Finally, we obtain the distance between p and s by computing the distance between p and p' with :

$$p' = n_1 + t.n_1n_2 \tag{6.10}$$

$$d_{min} = \sqrt{(x_p - x_{p'})^2 + (y_p - y_{p'})^2} \tag{6.11}$$

Afterward, segments that have a bigger distance than a fixed threshold distance $d_{thresh_{min}}$ are eliminated and the remaining are picked as Way candidates.

6.3.2.2/ ANGLE DIFFERENCE

As shown in Figure 6.5, we compute the angle difference between the GPS trace and the Way. We retain Ways that have an angle difference lower than 90°.

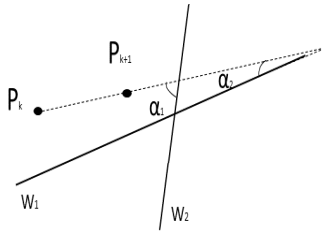


Figure 6.5: Difference of angles α_1, α_2 between GPS trace and Ways W_1, W_2

6.3.2.3/ SPEED DIFFERENCE

Quddus et al., [39] stated the MM algorithm has to take into account the road design parameters. In our work, the OSM provides meta-information about the Ways. One of this information consists of the speed limits which can be retrieved using the value for the tag *"MaxSpeed"*. Hence, we compare the speed information of the ego vehicle to the value of the "MaxSpeed" tag. If the speed of the vehicle is greater than a threshold then

the Way is eliminated. The threshold is defined as the sum of the limitation speed, from the tag *"MaxSpeed"*, plus a threshold, the value of this threshold allows the ego-vehicle to exceed the speed limit in a reasonable manner.

Once the discrimination stage is completed, if the number of candidate segments is higher than one, we proceed to the selection of the correct segment.

6.3.3/ Selection Stage

The previous criteria are not always sufficient to select the right segment. There might still be ambiguity on choosing the right segment.
In order to choose the right segment, we define the state of ego-vehicle at time t as $\mathbf{x}_t = (x, y, \theta)^T$. The selection procedure can be formulated as finding the highest conditional probability of belonging to a segment s_i knowing the pose of the ego-vehicle $\mathbf{x}$ at each time t:

$$\arg\max\ p(\mathbf{s}_i|\mathbf{x}_t) \tag{6.12}$$

The main goal of this formulation is to select the correct segment on which the vehicle is traveling. Consequently, a selection criterion must be introduced to determine the correct segment. As response to the equation (6.12), three probabilistic criteria [133] were developed:

- C_d: Criterion based on Euclidean distance.
- C_m: Criterion based on Mahalanobis distance.
- C_p: Criterion based on the probability of belonging to a segment.

6.3.3.1/ Probabilistic criterion based on Euclidean distance

This criterion uses only the GPS measurement in order to choose the correct segment and do not need the vehicle pose estimation. Therefore, for this criterion, the map matching task can be formulated as the calculation of the highest posterior probability of a GPS measurement z_k belonging to a segment s_i:

$$\arg\max_i\ p(s_i|z_k) \tag{6.13}$$

By assuming the uniformly distributed prior probability for the segments $p(s_i)$ and using Bayes formula [30], we get:

$$\arg\max_i p(s_i|z_k) = \arg\max_i(\frac{p(z_k|s_i)(s_i)}{p(z_k)}) \tag{6.14}$$

Since $p(z_k)$ is constant for each segment candidate, then we have:

$$\arg\max_i p(s_i|z_k) \equiv \arg\max_i\ (p(z_k|s_i)p(s_i)) \tag{6.15}$$

Since p(s_i) is uniformly distributed, we get:

$$\arg\max_i p(s_i|z_k) \equiv \arg\max_i\ p(z_k|s_i) \tag{6.16}$$

The probability $p(z_k|s_i)$ is modeled by two random independent variables :

- The minimum distance to a segment d_{min} (cf. Subsection 6.3.2.1) is modeled as zero mean, normally distributed random variable with standard deviation $\sigma_{d_{min}}$ described as follows:

$$\sigma_{dmin} = \epsilon \times DOP^1 \tag{6.17}$$

  Due to the uncertainty of the GPS data (between 2 and 6 m) and the OSM map (between 6-9 m [62]), the theoretical error in distance ϵ was chosen to covert the two uncertainties. In this work it has been fixed at 15 m.

- The angle difference θ_{diff} is modeled as zero mean, normally distributed random variable with standard deviation $\sigma_{\theta_{diff}}$ defined as being inversely proportional to the speed of the vehicle v:

$$\sigma_{\theta_{diff}} \propto \frac{1}{v} \tag{6.18}$$

As a consequence, we rewrite (6.16) as follows:

$$C_d = \arg\max_i p(s_i|z_k) = \arg\min_i \left(\frac{d^2(z_k, s_i)}{\sigma_d^2} + \frac{\theta^2(z_k, s_i)}{\sigma_\theta^2} \right) \tag{6.19}$$

Finally, the probabilistic criterion C_d is calculated for every way candidate, the Way having the smallest criterion is chosen among the Way candidates (segment candidate).
This selection criterion relies only on GPS measurement, it does not take into account the correlation between the position of the vehicle in x and y directions and the vehicle heading θ. In addition, the heading of the vehicle is calculated from the GPS trace, leading to an estimation that is dependent on the accuracy of the GPS receiver, which is metric. As a consequence, the estimation of the heading of the vehicle is not optimal.
In order to address these issues, one solution would be to take into account the proprioceptive information of the ego-vehicle in order to estimate its pose. This can be done using an extended Kalman-Filter (EKF). Moreover, the uses of the EKF will allow us to get the uncertainty matrix related to the pose of the vehicle.

6.3.3.2/ PROBABILISTIC CRITERION BASED ON MAHALANOBIS DISTANCE

In order to estimate the uncertainty related to the pose of the vehicle and depending on the sensors' information available, a Kalman filter is introduced. The primary objective of this filter is to estimate the uncertainty related to the pose of the vehicle, i.e., having an estimation of the covariance matrix, denoted as $\mathbf{\Sigma_x}$, $\mathbf{x}$ associate with the vehicle' pose at each time frame, denoted as $\mathbf{x} = (x, y, \theta)^T$.
Using the covariance matrix associated with the pose of the vehicle $\mathbf{\Sigma_x}$, we compute the Mahalanobis distance for each Way candidate. To this end, we calculate the orthogonal projections of the vehicle position onto every Way candidate, as illustrated in Figure 6.6.

- $\mathbf{x}_s = (x_s, y_s, \theta_s)^\top$ the vector representing the orthogonal projection on the segment .

[1] Dilution of precision

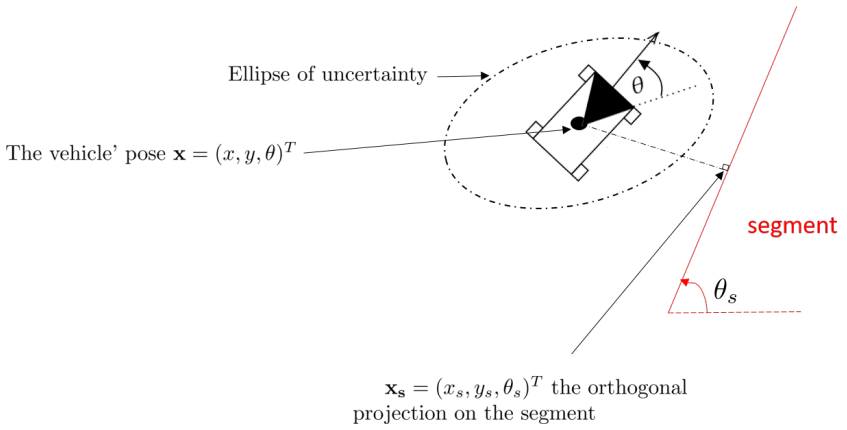


Figure 6.6: Example of calculation of the Mahalanobis distance. $\mathbf{x} = (x, y, \theta)^T$ the current state vector, and $\mathbf{x}_s = (x_s, y_s, \theta_s)^\top$ is the vector representing the orthogonal projection on the segment.

Afterward, the criterion based on the Mahalanobis distance C_m is calculated as follows:

$$C_m = \sqrt{(\mathbf{x} - \mathbf{x}_s)^T \Sigma_{\mathbf{x}}^{-1} (\mathbf{x} - \mathbf{x}_s)} \tag{6.20}$$

With the difference between the normalized angles $\theta \in [0, 2\pi)$ and $\theta_s \in [0, 2\pi)$ noted Δ_θ is calculated as follows:

$$\Delta_\theta = min(2\pi - |\theta - \theta_s|, |\theta - \theta_s|) \tag{6.21}$$

The segment with the smallest C_m is then chosen.

6.3.3.3/ Probabilistic criterion based on the probability of belonging to a segment

The idea behind this probabilistic criterion is to compute the probability of a segment $\mathbf{s}_i = (\mathbf{n}_{1i}, \mathbf{n}_{2i})^\top$ defined by $\mathbf{n}_{1i} = (x_1, y_1)^\top$ and $\mathbf{n}_{2i} = (x_2, y_2)^\top$ to belong to the expected area of the presence of the ego-vehicle. This area is delimited by the covariance matrix $\mathbf{\Sigma_x}$. To do so, first, we envelop the segment with an ellipse defined by a Gaussian of dimension n (as illustrated in Figure 6.7). To dimension this ellipse, we rely on the canonical representation of Gaussian to compute its covariance matrix $\mathbf{\Sigma}_s$. As result the covariance matrix $\mathbf{\Sigma}_s$ is defined as follows:

$$\mu_s = \begin{pmatrix} (x_2 + x_1)/2 \\ (y_2 + y_1)/2 \end{pmatrix} \tag{6.22}$$

$$\mathbf{\Sigma}_s = \mathbf{VLV}^{-1} \tag{6.23}$$

$$\mathbf{V} = \begin{pmatrix} x_2 - x_1 & y_1 - y_2 \\ y_2 - y_1 & x_2 - x_1 \end{pmatrix} \tag{6.24}$$

$$\mathbf{L} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} \frac{|s|^2}{4} & 0 \\ 0 & \frac{Th_s^2}{4} \end{pmatrix} \tag{6.25}$$

With Th_s being the thickness of every segment, λ_1 and λ_2 the eigenvalues of the matrix $\mathbf{V}$. Thus, the probability we are looking for is defined as follows:

$$C_p = \frac{1}{p}\exp(g_1 + g_2 + \frac{1}{2}\mu^T\mathbf{\Sigma}^{-1}\mu) \tag{6.26}$$

with:

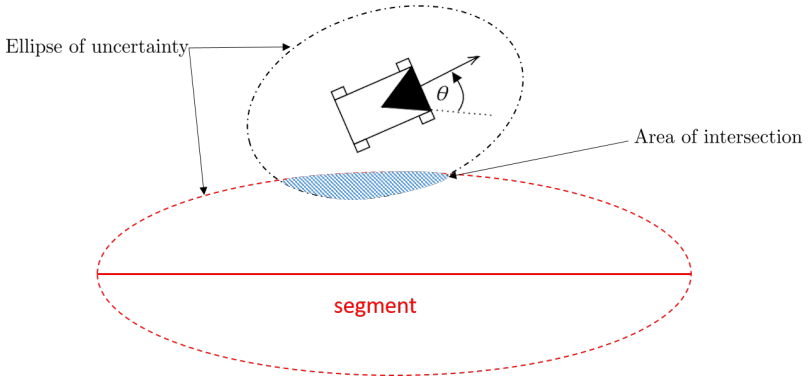


Figure 6.7: Illustration of the probability of the segment (in red) to belong to the expected area of presence of the ego-vehicle, the blue area corresponds to the probability we are looking to compute.

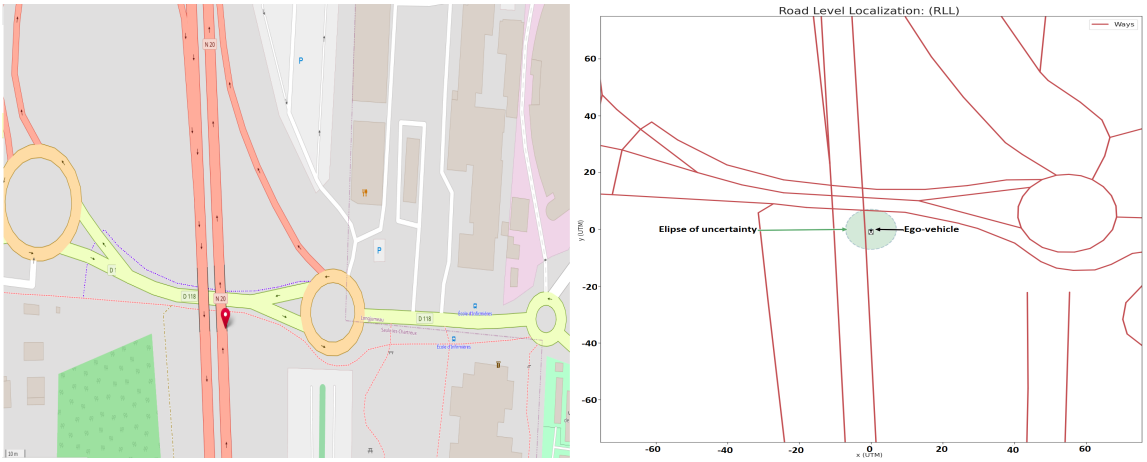
- $\mathbf{\Sigma} = [\mathbf{\Sigma_x}^{-1} + \mathbf{\Sigma_s}^{-1}]^{-1}$
- $\mu = [\mathbf{\Sigma_x}^{-1} + \mathbf{\Sigma_s}^{-1}]^{-1}[\mathbf{\Sigma_x}^{-1}\mathbf{x} + \mathbf{\Sigma}_s^{-1}\mu_\mathbf{s}]$
- $p = \log\left[(2\pi)^{-n/2}|\mathbf{\Sigma}|^{-1/2}\right]$
- $g_1 = \log\left[(2\pi)^{n/2}|\mathbf{\Sigma_x}|^{-1/2}\right] - \frac{1}{2}\mathbf{x}^T{\mathbf{\Sigma}_\mu}^{-1}\mathbf{x}$
- $g_2 = \log\left[(2\pi)^{n/2}|\mathbf{\Sigma_s}|^{-1/2}\right] - \frac{1}{2}\mu^T{\mathbf{\Sigma_s}}^{-1}\mu$

The details of the computation are presented in Annex 10.3.
The segment with the highest probability C_p is chosen.

6.3.4/ FUNCTIONING OF THE ALGORITHM

In the following, a canonical example that helps to understand the functioning of the algorithm is illustrated. In all figures, the vehicle is represented by a rectangle box, and the uncertainty about its position is represented with a green ellipse. All the steps of the PMMA are given and are accompanied by graphical illustrations.

Initialization stage In this initial stage, the OSM data around the measured GPS position of the vehicle are retrieved as illustrated in Figure 6.8a The data retrieved are in the form of Ways. Each Way can be either a road or other geographical road object. The Ways collected from the OSM database are represented in red in the Figure 6.8b.



(a) (b)

Figure 6.8: Initialization step for the MM algorithm. Red lines represent the Way extracted from OSM.

Preprocessing The global aim of this module is to reduce the complexity problem of the MM by eliminating Ways that do not represent roads on which the vehicle can potentially travel. In other words, eliminate Ways that have not the "Highway" tags value. The results of this preprocessing stage are illustrated in Figure 6.9. As we can see Ways that do not represent roads (as illustrated in Figure 6.8a) are eliminated.

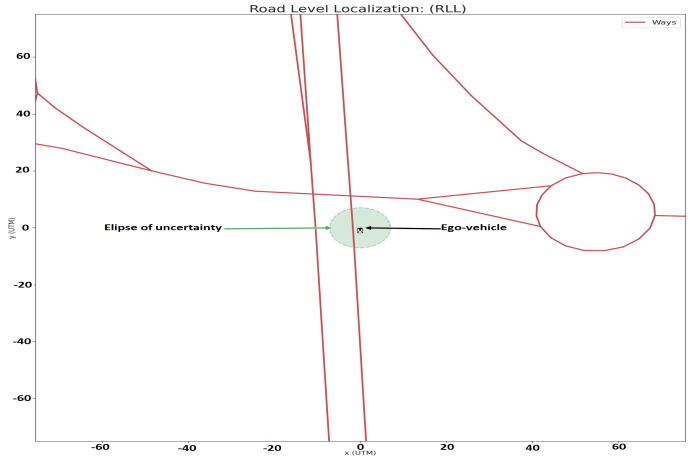


Figure 6.9: Preprocessing step with the elimination of roads that are not suitable for MM.

Discrimination stage Now that only Ways representing roads remain, the objective of the discrimination stage is to eliminate ways that are not suitable to perform the MM. In other words, Ways that do not correspond to the vehicle's configuration. Therefore, we eliminate Ways depending on two factors: the distance to the estimated position of the vehicle as illustrated in Figure 6.10(a) and the angle difference between the Way and the vehicle heading as illustrated in Figure 6.10(b). The remaining Ways are considered as compatible with the vehicle's configuration.

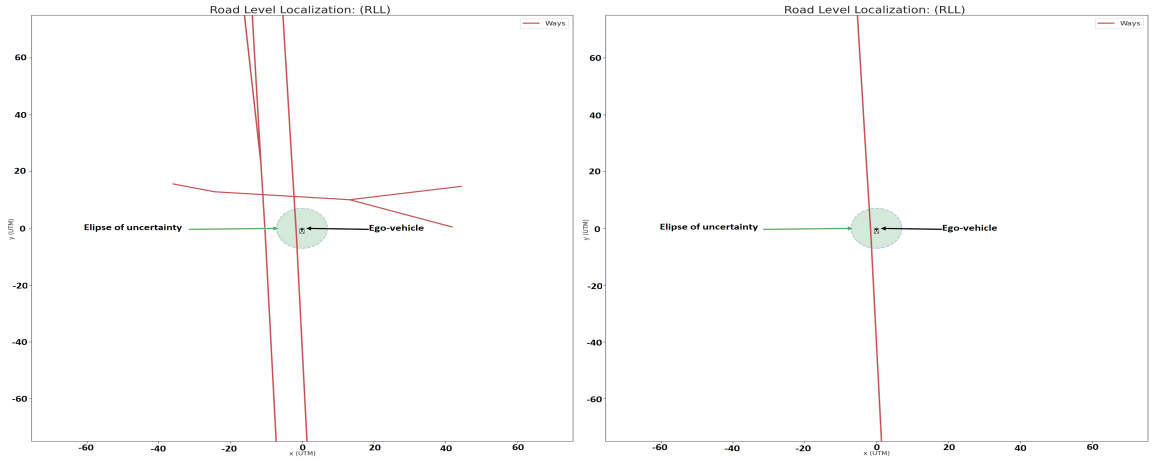


Figure 6.10: Discrimination stage depending on the distance (a) and the orientation of the vehicle (b).

Selection stage Depending on the number of Ways candidate remaining, the selection is performed using one of the criteria introduced in the preceding section as illustrated in Figure 6.11

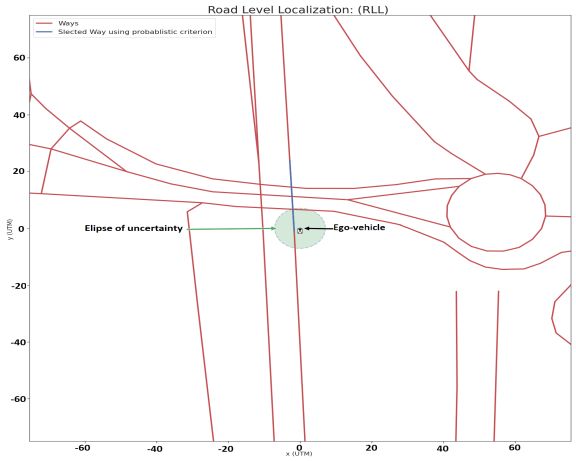


Figure 6.11: Selection of the right Way in blue.

6.3.5/ THE ENHANCED PROBABILISTIC MAP MATCHING ALGORITHM (PMMA)

The PMMA proposed in the previous sections does not take into consideration the navigation history and the topology connection that may exist between Ways. A solution to take into account these considerations is to model the MM task using a Hidden Markov Model (HMM).
By definition a HMM consists of two stochastic processes, the first one is a Markov Chain to model the change of a state vector over time. This change is governed by a probability that describes the "transition probability" over time. In the HMM, the states of the chain are not visible but observable, for this reason, they are called “hidden”. The second

process is called the "observation space", it produces the emission of the observation at each time. Although the elements of the state vector are hidden, there is a relation between the hidden elements of the state and the observations, this relation is referred to as an emission probability. Figure 6.12 describes the proposed HMM for the map-matching task.

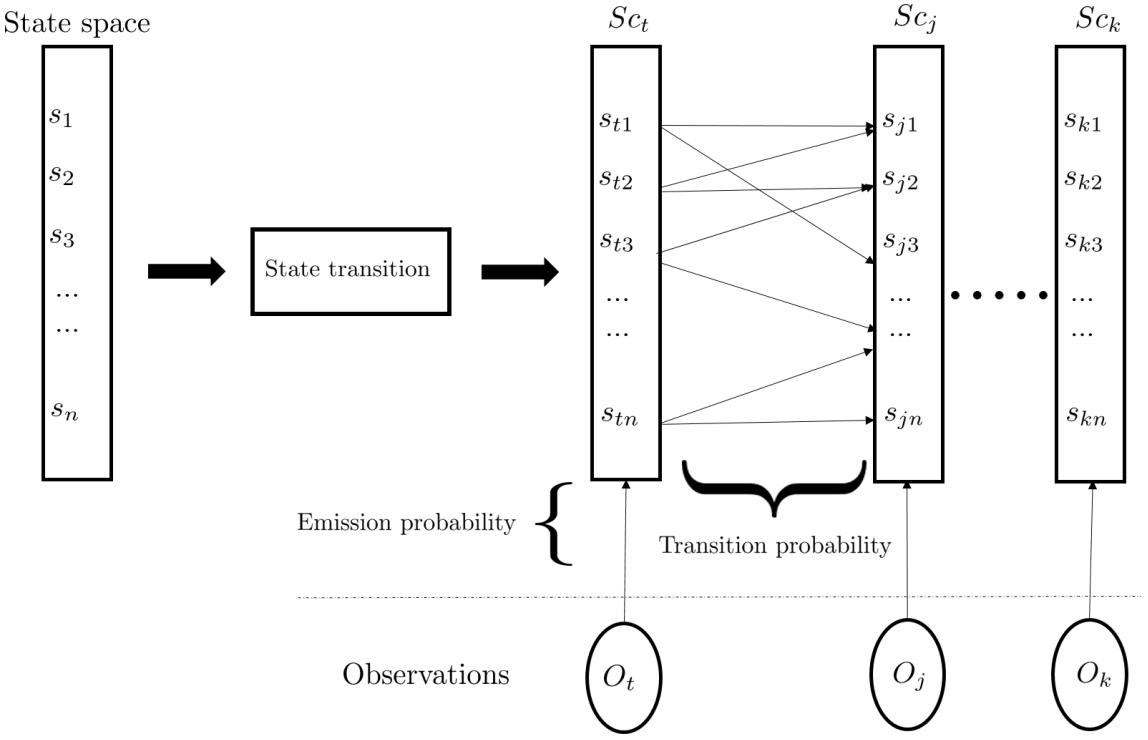


Figure 6.12: Modeling of the Hidden Markov model Map-Matching (HMM-MM) algorithm, with Sc_t the state space at time t, s_t the segment candidate at time t and O_t the observations at time t. Note that the number of candidate segments at each time may vary.

Accordingly, in order to model the HMM, three components must be defined:
State space: the state of the system describes the list of the candidate segments for each observation. We will use Sc_t to denote the set of candidate segments at time t $Sc_t = \{s_1, s_2, ..., s_{n_t}\}$ with n_t the number of candidate segments at each time t.
Observation space: for each candidate segment composing the state space, an emission probability is made. This emission probability is directly calculated from the multi-criteria algorithm. For each segment candidate s_i the emission probability $P_e(s_i)$ is calculated as follows:

$$P_e(s_i) = \frac{C_k(s_i)}{\sum\limits_{j \in Sc_i} C_k(s_j)} \tag{6.27}$$

where C_k is one of the probabilistic criterion used with $k \in \{d, m, p\}$.
Transition probability: the transition probability reflects the probability that a state will move from one state to another. In the map matching procedure, the topology of the network is used in order to determine the transition matrix. Indeed, the vehicle can only move on Ways that are physically connected.
We will refer to $P_t(s_i^t, s_j^\tau)$ as the transition probability from a segment s_i^t to a segment s_j^τ

Table 6.2: Transition matrix for example shown on Figure 6.13

	s_1	s_2	s_3	s_4	s_5
s_1	λ^s_{11}	λ^n	λ^c_{13}	λ^n	λ^c_{15}
s_2	λ^n	λ^s_{22}	λ^n	λ^n	λ^n
s_3	λ_{31}	λ^n	λ^s_{33}	λ^c_{34}	λ^c_{35}
s_4	λ^n	λ^n	λ^c_{43}	λ^s_{44}	λ^n
s_5	λ^c_{51}	λ^n	λ^c_{53}	λ^n	λ^s_{55}

given the state space $S\,c_t$ and $S\,c_\tau$ for time t and τ:

$$P_t(s_i^t, s_j^\tau) = \frac{\lambda_i}{\sum\limits_{j \in S\,c_t} \lambda_j} \tag{6.28}$$

Where λ_i is a criterion calculated for each segment, this criterion obeys some rules defined as follows:

1. $\lambda_i = \lambda^n$ if the segment s_i^t and s_j^τ are not connected;
2. $\lambda_i = \lambda^s$ if the segment s_i^t and s_j^τ are the same;
3. $\lambda_i = \lambda^c$ if the segment s_i^t and s_j^τ are connected.

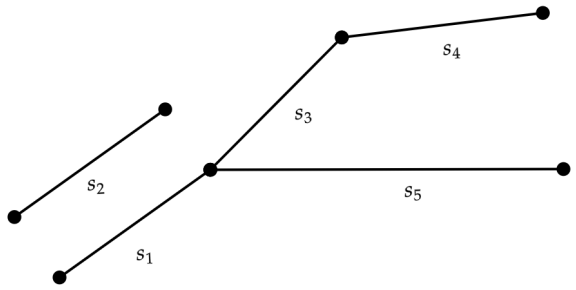


Figure 6.13: Example of road network to illustrate the transition probability calculation. In this example the number of Way candidates is the same between two consecutive frames

Figure 6.13 shows a simplified road network to illustrate the transition matrix probability. These probabilities are shown in Table 6.2. It can be noticed that in this example the number of Ways is the same for two connected states, which is not always the case. Indeed, the number of Ways may vary from two consecutive frames.

- The transition probability λ^n is used if the two segments are not connected, in this case the probability of transition is equal to zero. Indeed, the vehicle can not travel if two segments are not connected. Naturally, this assumption holds only for small-time interval between two measurements.
- The transition probability λ^s is used whenever the two segments are the same, in other words, they belong to the set $S\,c_t$ and $S\,c_{t+1}$. Let $\mathbf{P} = (x, y)^T$ be the vector describing the Cartesian coordinates of the ego-vehicle in the Universal Transverse

Mercator coordinate system frame (UTM), we note $\mathbf{P'}$ the projection of the point $\mathbf{P}$ on the segment and ψ the segment's heading, the segment s is composed of two nodes n_1, n_2. One can note that the computation of the point $\mathbf{P'}$ follows the same strategy adopted in the Subsection 6.3.2.1.
The probability to change the segment will be smaller if the vehicle is located in the middle of the segment, and vice versa it will be greater if the vehicle is within the limits of the segment. Mathematically this probability is linked to the Gaussian distance between the point P' and the middle of the segment $\mathbf{s}$ as illustrated in figure 6.14. Thus, it is calculated as follows:

$$\lambda^s = e^{-\frac{1}{2}\left(\frac{t-u_0}{\sigma_0}\right)^2} \tag{6.29}$$

With $P' = t.n_1n_2$, $u_0 = 0$ and we take $\sigma_0 = 0.25$. This value allows us to cover the middle of the segment, without covering the ends of the segment.

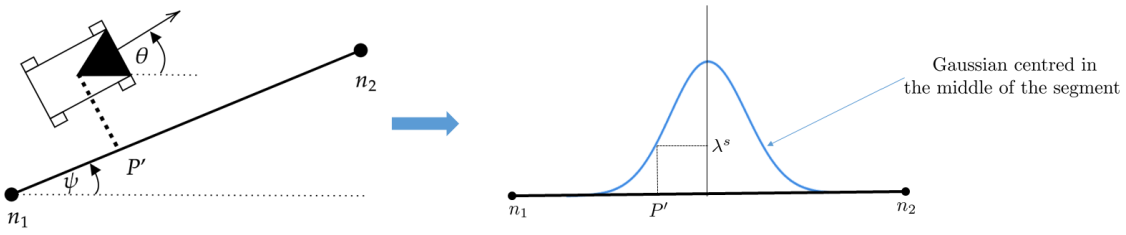


Figure 6.14: Illustration of calculation for transition probability λ^s, the vehicle is illustrated by the box with black triangle, ψ represents the segment heading, θ the vehicle's heading, and $\mathbf{P'}$ is the projection of the vehicle position on the segment $s = [n_1, n_2]$.

- The transition probability λ^c is used whenever the segment s_i and s_j are connected. In this case we use a similar criterion to the one presented by lue *et al.*[117]. This criterion depends on the difference between the vehicle's heading and the segment's heading :

$$\lambda^c = e^{-\beta|\Delta\theta - \Delta\psi|} \tag{6.30}$$

with $\Delta\theta$ being the vehicle's heading change over time, $\Delta\psi$ the segment's heading change between the two segments s_i and s_j and β a chosen coefficient.

Now that the observation model and the transition model are set, the skeleton of the proposed HMM is completed, and therefore the MM procedure can be performed.
In the following section, experimental results are given for the functioning of the PMMA

6.4/ RESULTS AND DISCUSSION

In order to show the effectiveness of our proposed solution, real-world experiments have been conducted. To this end, a dataset contains GPS data frame were collected in the region of Paris for a total of 6596 GPS frames. The GPS receiver used for the experiment provides a decametric accuracy measurement of the longitudinal and lateral coordinates.

Furthermore, signal losses occur several times during the experiment. The GPS data are received with a frequency of 1Hz. The ground truth is obtained from the video of a camera installed in front of the vehicle, the route traveled is shown in the Figure 6.15.
In order to assess the presented RLL algorithm, the first part of the discussion will focus

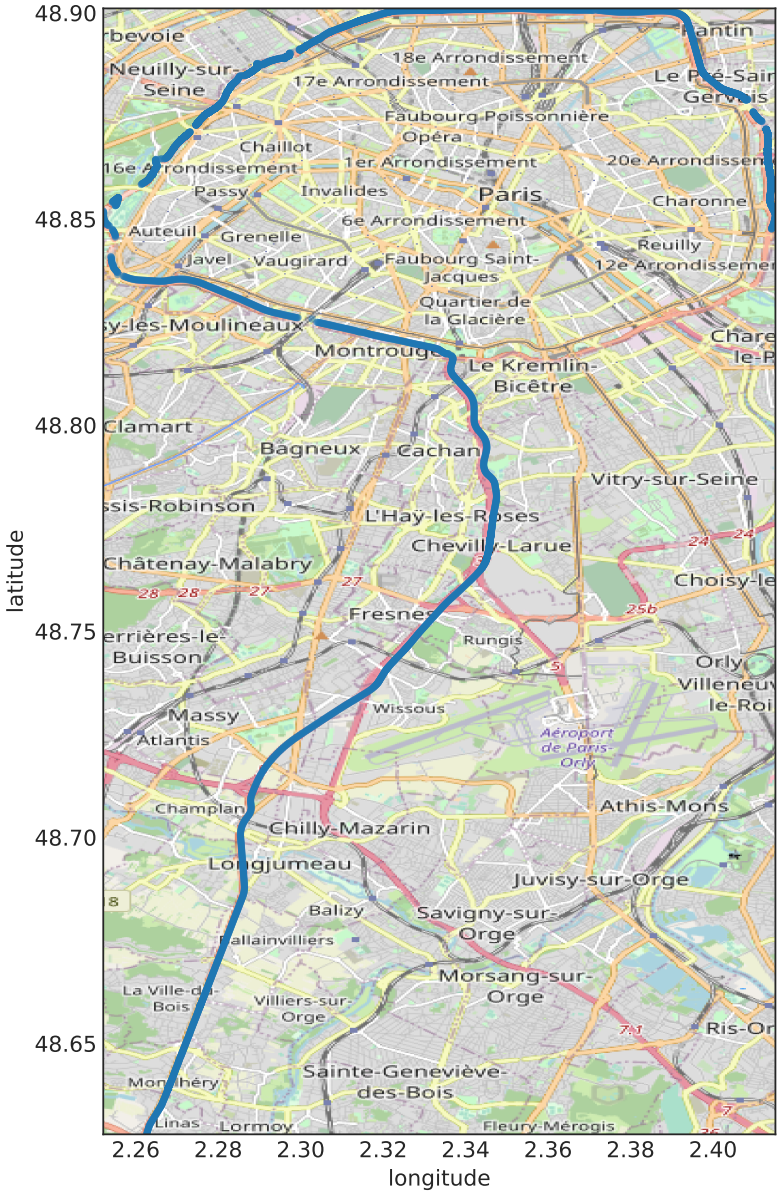


Figure 6.15: The route traveled in the region of Paris illustrated with blue color in OSM. Around 100 km have been traveled for a total of 6596 GPS frames.

on comparing the MM accuracy provided by each criterion defined in Section 6.3.3(C_d, C_m, and C_p), then we test the enhanced version of the algorithm (with the addition of the HMM). The latter will be presented in Chapter 9.
Nevertheless, the comparison between the three criteria (C_d, C_m, and C_p) will be done regarding the lanes number estimation. Indeed, the nature of the dataset used (the decametric accuracy of the GPS and the loss of signal) makes it hard to manually chose the

correct way in which the vehicle is traveling. As consequence, the number of lanes estimation results are summarized in Table 6.3 and illustrated in Figure 6.16. The incorrect number of lanes estimations are divided into two groups. The first one "Zero estimations" denotes the absence of information about the number of lanes from the OSM geodata. In this case, the algorithm provides zero for lane number estimation. The second one "Wrong estimations" indicates the false estimations that are different from the "zero estimation" group.

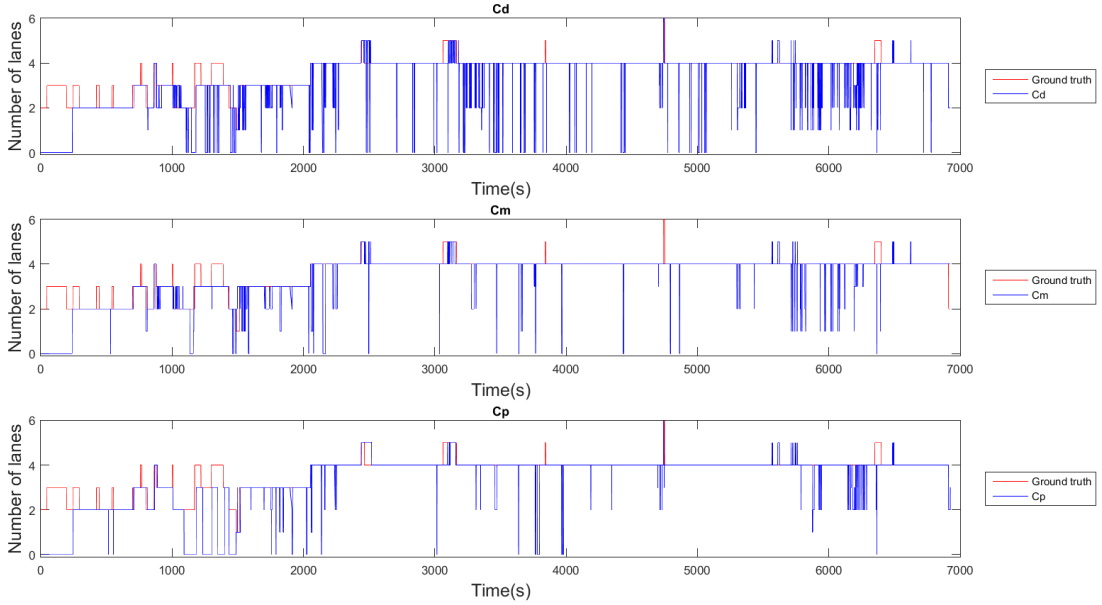


Figure 6.16: Lanes estimation compared to ground truth for every criterion used in the map matching [133].

In order to prove it, we run the same algorithm with the same data sets on a different period (April 2018). During the time interval between the two runs, the OSM map has been updated, and therefore the map should provide more information about the lanes number. The results obtained in Table 6.4 support our argument, as we expected the zero estimations has decreased for each criterion used. In addition, the wrong estimations have also decreased, meaning that the wrong estimations are not only due to a wrong map matching. Based on these results, we believe that there are three reasons for the lanes number estimation to be false:

Table 6.3: Results for the accuracy rate on lanes number estimation for every criterion (February 2018)

	C_d	C_m	C_p
Correct estimations	79%	82.95%	81.58%
Wrong estimations	13.89%	11.37%	9.72%
Zero estimations	7.11%	5.68%	8.70%

- The map matching is correct but the tag "lanes" does not exist for the selected Way (the zero estimation)
- The map matching is correct but the information "number of lanes" is wrong or has not being updated.
- The map matching is false because the wrong Way was selected

For the first and second cases, the results of the lanes number estimation depend on the quality of the OSM map as explained in the previous paragraph. However, it is more difficult to distinguish between the second and the third cases when the number of lanes estimated is wrong. Nevertheless, after investigating situations where our lanes number estimation is incorrect because of a wrong map matching, we figured out that these cases happen when an exit lane occurs as illustrated in Figure 6.17. In these situations, having the history navigation of the selected Ways candidate will ultimately result in having better accuracy in choosing the right segment.
To sum up, this experiment leads to two major conclusions, the first one is that the criterion based on the Mahalanobis distance C_m is more robust than the other two criteria. Indeed, for this criterion, no threshold parameters are needed. The second conclusion is that it is necessary to keep the history navigation of the MM procedure. Therefore, results with the addition of the HMM will be presented in Chapter 9.The work presented in this chapter have been published in the IEEE International Conference on Intelligent Transportation System (ITSC18) [133].

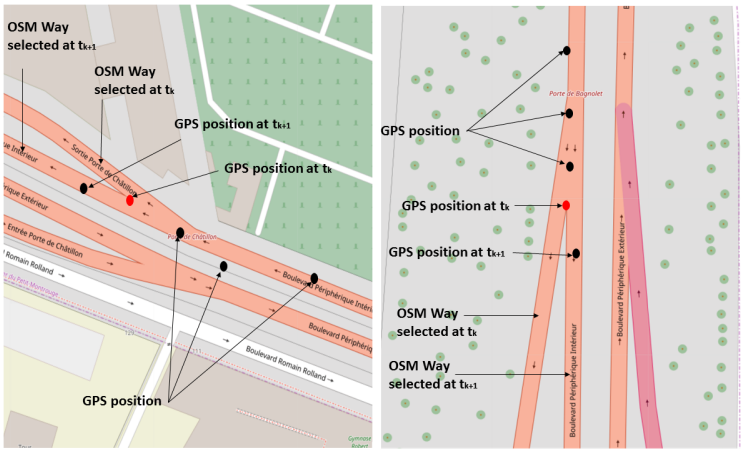


Figure 6.17: Scenarios where the map matching algorithm selects the wrong segment and leads to a false lanes number estimation

Table 6.4: Results for the accuracy rate on lanes number estimation for every criterion (April 2018)

	C_d	C_m	C_p
Correct estimations	79.85%	83.64%	83.14%
Wrong estimations	13.18%	11.47%	8.51 %
Zero estimations	6.97%	4.89%	8.35%

6.5/ CONCLUSION

In this chapter, a Road Level Localization (RLL) solution is proposed based on Probabilistic Map Matching Algorithm (PMMA). This architecture assembles several interconnected designed modules that are responsible for MM procedure while taking into account uncertainties that comes from raw inputs sensors data. The PMMA is mainly composed of two blocks.
In the first block, a probabilistic framework is designed to match the correct road on which the vehicle is traveling. A first module is responsible for eliminating all the roads which are not compatible with the vehicle configuration in terms of distance and orientation. In addition, road design parameters, such as turn restrictions, speed limitation have been taken into account in this discrimination stage. Following this stage, a selection strategy based on three probabilistic criteria have been used. The mathematical formalizing of each criterion have been justified. Finally, comparison between criteria in terms of methodology and in terms of results have been given, leading to the conclusion that this probabilistic framework is not sufficiently robust to deal with ambiguous scenarios.
Furthermore, based on the output of the first block a second complementary block have been introduced in order to clear up ambiguous situations. It benefits from the architectural strength of the HMM in order to model the road topology and the vehicle's cinematic. The modeling is accomplished by defining transition and emission state probabilities that match the road topology while taking into account of the vehicle's cinematic.
In addition to the details given about the main blocks and their interactions composing the proposed PMMA, experimental results have been conducted. The proposed algorithm have been tested on a dataset and results were shown in this chapter to validate the overall architecture. Further results concerning this part will be given in Chapter 9.
In the next chapter, the second part of the proposed end-to-end localization framework is introduced. It corresponds to the Ego-Lane Level Localization (ELL) algorithm proposed in this P.h.D work.

7

EGO-LANE LEVEL LOCALIZATION SOLUTION

Contents

7.1 Problem formulation 103
7.2 Proposed solution 104
7.3 Recursive Information-Driven Algorithm (RIDA) 105
7.3.1 Initialization stage 105
7.3.2 Selection of the best Region Of Interest (ROI) 109
7.3.3 Detection 112
7.3.4 End 112
7.3.5 Tracking step 113
7.3.6 Canonical example 114
7.4 Results and discussion 117
7.4.1 The datasets 117
7.4.2 Construction of the dataset 117
7.4.3 Quantitative results 118
7.4.4 Qualitative results 119
7.5 Conclusion 120

7.1/ PROBLEM FORMULATION

Ego-Lane Level Localization (ELL) has been an active field of research due to the importance of this localization knowledge in the majority of the ADAS applications [121] like Lane-Departure-Warning Systems, Adaptive Cruise Control, and Lane Change Assist. For this kind of ADAS application, knowing the position of the ego-vehicle with respect to its lane is mandatory. As a consequence, an ELL algorithm that guarantees this requirement is a critical part of any localization system. By and large, the state-of-the-art review conducted in Chapter 3 leads to the conclusion that this task is carried out primarily through the identification of ego-lane markings. In recent years, ego-lane marking detection systems have been the subject of various research topics, using several input data such as camera or lidar sensors. However, the majority of the

discussed techniques are limited to image mask retrieval and do not provide a usable representation of the detected lane marking in world coordinates. Hence, they are unfit to perform the ELL. In addition to that, an ELL algorithm based on ego-lane marking detection has to fit the detected patterns to a model that suits the road's shape. However, all the roads do not share the same skeleton, and finding a unified road model is still an ongoing field of study. On the other hand, highway road models have been widely studied, and mathematical models that suit the road's shape have been proposed. In that sense, when dealing with highway roads, the ELL solution has to take into account the global shape of the roads in its detection process.
Under the light of these considerations, we propose an information-driven method for ELL that takes into account inaccurate prior geometry of the road from OSM to perform ego-lane marking detection. In addition to that, the method allows a world coordinate representation that can be used for ADAS application. Furthermore, the proposed method works on both lidar and camera images.

7.2/ PROPOSED SOLUTION

Once the RLL has been performed, the Ego-Lane Level Localization (ELL) is initiated. We choose to start the ELL before the Lane-Level Localization (LLL) for several reasons: the first is that the ego-lane marking is, for the most part, the easiest one to detect. The second one is due to the use of the information-driven approach. Indeed, when the ELL is completed we have an estimate of the ego vehicle's lateral position in its lane. So, knowing the lane's width and the lane's number allows us to interpolate research zones for other lane marking.
In that regard, we present the Recursive Information-Driven Algorithm (RIDA) that works on both camera and lidar data. In contrast to what is described in most of the works available for ego-lane detection, we present an informational-driven approach that is inspired by the work of [18] for camera images. The concept is based on a focusing algorithm to detect the ego-lane marking. To do so, we take advantage of a probabilistic model that captures the road's shape prior information coming from OSM in order to limit the search areas for the ego-lane marking detection process. Once the search area is created, we divide this zone into n_{roi} Region Of Interests (ROI) that has the same height. Afterward, we propose a selection strategy based on a Bayesian Network (BN) coupled with the entropic features to choose the most informative ROI, which means the ROI where we have the best chance to detect a pattern (a segment) of the ego lane marking. After each detection, we update the probabilistic model which leads to a new searching area, and accordingly reduces the size of the ROI.
One of the main advantages of this method appears in the case of the wrong detection of a segment. This wrong detection will produce an update to a wrong probabilistic model. Hence, the resulting ROI will no longer cover the ego-lane marking. If this scenario happens, the recursive nature of the proposed method allows us to downgrade to the previous probabilistic methods (before detecting the wrong segment). These steps are repeated until an entropic criterion is achieved indicating that the ego-lane marking has been detected.
This algorithm has been firstly introduced by Aufrère *et al.* [18] in order to detect lane marking using a camera. However, we propose an enhanced version of this algorithm with the addition of the entropic selection strategy. Furthermore, we extend its uses to the

lidar data. Globally, the general architecture of the proposed solution remains the same with slight change due to the nature of the data. The overall pipeline of the proposed architecture is shown in Figure 7.1, it follows the global architecture of the model-driven approaches defined in Figure 3.2. Therefore, it is composed of the following parts:

- **Initialization stage:** in this stage, initialization of the road model is performed.
- **Selection of the best ROI:** the objective of this stage is to select the most informative ROI in terms of accuracy and probability of successful detection
- **Detection:** in this stage a detection of a primitive is performed in the selected ROI.
- **Update:** depending on the result of the detection, the model is updated and the confidence in the global model is also updated.
- **End:** to end the recognition process, we define an entropic criterion that must be satisfied.

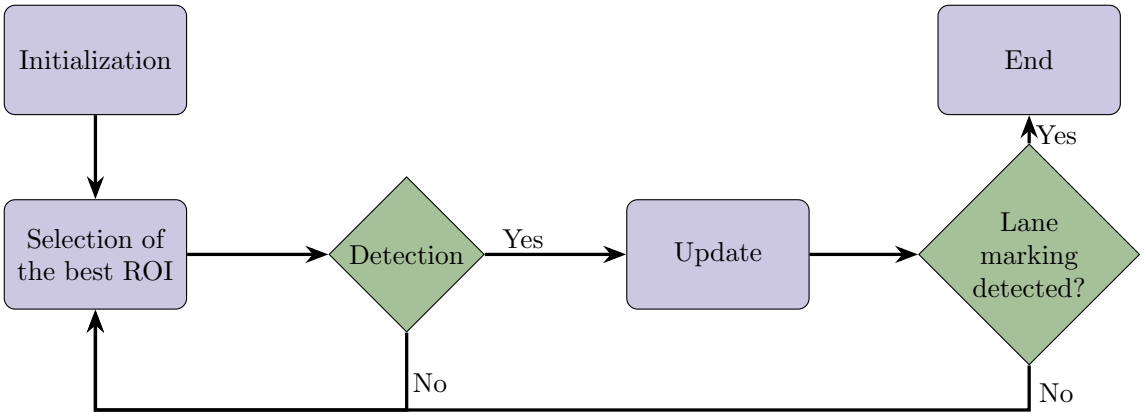


Figure 7.1: All different steps of the Recursive Information-Driven Algorithm (RIDA) for ego-lane marking detection.

7.3/ RECURSIVE INFORMATION-DRIVEN ALGORITHM (RIDA)

In the following sections, details about the aforementioned parts are given. In addition to that, a parallel between the functioning of the algorithm on both lidar data and camera images is given. The objective is to illustrate the difference in the functioning of the RIDA between the two sensors.

7.3.1/ INITIALIZATION STAGE

In this stage, the road model is transposed in the sensor frame (lidar or camera). However, before it must be modeled on the vehicle's frame (x_v, y_v). Contrary to the work of Alvarez *et al.*[83], only the skeleton of the road will be projected in the vehicle's frame. Indeed,

the objective is to detect the lane marking and not the entire road (asphalt). Therefore, the road model is defined as a cubic polynomial [108][119] as shown on Figure 7.2

$$y(x) = \frac{1}{6}\, c'\, x^3 \; + \frac{1}{2}\, c_0\, x^2 \; + \psi\, x \; + y_0 \tag{7.1}$$

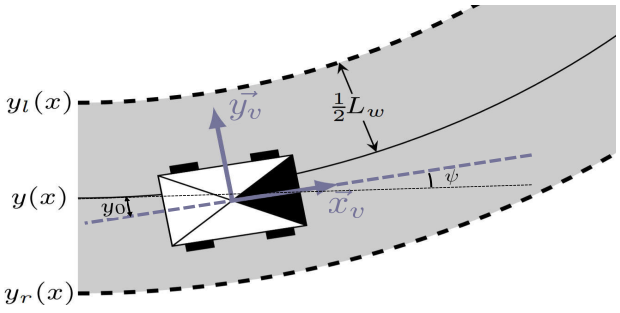


Figure 7.2: The robot navigates on the lane of width L_w modeled as a polynomial equation $y(x)$. The left and right borders of the lane are respectively approximated with the polynomials $y_l(x)$ and $y_r(x)$.

The parameter c' is the curvature's derivative of the road, c_0 the curvature of the road, ψ the vehicle's heading with respect to the tangent of the road, y_0 the lateral shift of the ego-vehicle in relation to the road model. For a lane of width L_w, the left or right ego-lane markings denoted by $y_{l,r}$

$$y_{l,r}(x) = y(x) \pm \frac{1}{2}\frac{L_w}{\sqrt{1 + y'(x)^2}}, \tag{7.2}$$

where $y'(x) = \frac{\mathrm{d}}{\mathrm{d}x}y(x)$. One can note that the equation for the markings is not polynomial. However, as the curvature of the road is small on the highways and most marked roads, the derivative $y'(x)$ can be neglected for small x. The equation simplifies to

$$y_{l,r}(x) \approx y(x) \pm \frac{1}{2}L_w \tag{7.3}$$

In practice, the approximation even holds for large x, being accurate enough up to 50 m. Furthermore and as discussed in Section 6.2, the OSM map does not provide information about the accuracy of its data. However, it is well known that OSM is a collaborative project in which volunteers provide the geospatial data. If we consider that most of the volunteers have a classic GNSS receiver, the accuracy of the OSM is thus metric.
To encompass these uncertainties for each ego-lane marking, we define a probabilistic model composed of a state vector $\mathbf{x}_{l,r}$ and its associated covariance matrix $\mathbf{\Sigma}_{\mathbf{x}_{l,r}}$. The state vector $\mathbf{x}_{l,r}$ contains the mean values of the parameters of the roads presented in Equation 7.1. The objective of this state vector is to capture the priors about the road's shape. Therefore, it is defined as follows:

$$\mathbf{x}_{l,r} = [\mu_{c'},\ \mu_{c_0},\ \mu_{\psi},\ \mu_{y_0}, \mu_{L_w}]^T, \tag{7.4}$$

with μ refereeing to a mean value. The values $(\mu_{c'},\ \mu_{c_0})$ are taken from OSM. The polynomial representation of the road (Equation 7.1) will be affected by the unknown accuracy

of the OSM data. For the parameters c' and c_0, they represent the shape of the road. Given that we are working mostly with highway roads, their values will not be affected by the inaccurate geospatial accuracy of the OSM dataset. Hence, their value can be used directly from Equation 7.1. In contrast, the parameters ψ, y_0, L_w are more sensitive to inaccuracy of the OSM, and its value will be affected. That being said, the values $\mu_\psi, \mu_{y_0}, \mu_{L_w}$ are initialized using global knowledge about the road configuration and the vehicle pose. Globally, a vehicle has a tendency to drive in the center of its lane and its heading is parallel to the tangent of the road. Concerning the width of lanes L_w, depending on the legislation of the country, this value is set for all the roads that fall in the same category (i.e., 3.5 m in France for highway).
As a consequence, these parameters are used as priors to define the vector $\mathbf{x}_{l,r}$, and are initialized as follows

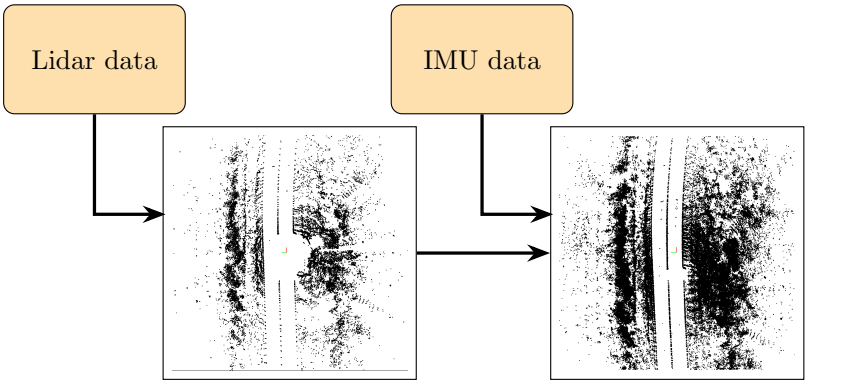
$$\begin{aligned} \mu_{c'} &= c' \\ \mu_{c_0} &= c_0 \\ \mu_\psi &= 0 \\ \mu_{y_0} &= 0 \\ \mu_{L_w} &= 3.5m \end{aligned}$$

The inaccuracy of the values contained in the vector $\mathbf{x}_{l,r}$ are counterbalanced using the co-variance matrix $\mathbf{\Sigma}_{\mathbf{x}_{l,r}}$ which is initialized as

$$\mathbf{\Sigma}_{\mathbf{x}_{l,r}} = \begin{pmatrix} \sigma^2_{c'} & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{c_0} & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{\psi} & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{y_0} & 0 \\ 0 & 0 & 0 & 0 & \sigma_{L_w^2} \end{pmatrix}. \tag{7.5}$$

σ^2 indicates the variance associated with each parameter of the vector $\mathbf{x}_{l,r}$. This matrix expresses the allowed dispersion around these parameters. The greater these σs are the less accurate the values of vector $\mathbf{x}_{l,r}$ are.
Now that the probabilistic model $(\mathbf{x}_{l,r}, \mathbf{\Sigma}_{\mathbf{x}_{l,r}})$ set, we want to see how this model is reflected on the sensor frame. In other words, we want to compute the covariance matrix associated with the transformation of the polynomial representation in the sensor space.

Lidar Data Before the initialization of the probabilistic road model in the lidar space, a preprocessing stage is performed. Indeed, the lidar frames are rich in information. However, one single lidar frame is not sufficiently dense to correctly perform the ego-lane recognition. To overcome this drawback, we integrate the previous lidar frames into one frame based on the odometry information of the vehicle. For this work, we took the last 5 frames into account. Figure 7.3(a) and Figure 7.3(b) show the difference in the spareness of the lidar pointclouds.
We apply then a threshold on reflectivity. Since we are looking for the ego-lane marking, points cloud with high reflectivity will be chosen. Thereafter, we transform the probabilistic



(a) Pointcloud (b) Accumulated Pointcloud

Figure 7.3: Overall algorithm for the proposed ego-lane detection. (a) The lidar provides pointclouds. (b) Using an IMU, several pointclouds are accumulated.

model $(\mathbf{x}_{l,r}, \mathbf{\Sigma}_{\mathbf{x}_{l,r}})$ on the lidar frame in order to get the uncertainty about the cubic polynomial. To do so we compute the Jacobian matrix $\mathbf{J}_{y_{l,r}}$ of the polynomials $\mathbf{y}_{l,r}(x)$. Hence, the covariance matrix associated to the vector space $\mathbf{x}_{l,r}$ in the lidar space is computed as follows:

$$\mathbf{\Sigma}_{y_{l,r}} = \mathbf{J}_{y_{l,r}} \mathbf{\Sigma}_{\mathbf{x}_{l,r}} \mathbf{J}_{y_{l,r}}^{\top} \tag{7.6}$$

With:

$$\mathbf{J}_{y_{l,r}} = \begin{pmatrix} \frac{\partial y_{l,r}}{\partial c'} & \frac{\partial y_{l,r}}{\partial c_0} & \frac{\partial y_{l,r}}{\partial \psi} & \frac{\partial y_{l,r}}{\partial y_0} & \frac{\partial y_{l,r}}{\partial L_w} \end{pmatrix} \tag{7.7}$$

The resulting model $(\mathbf{y}_{l,r}, \mathbf{\Sigma}_{\mathbf{y}_{l,r}})$ is transformed into the lidar space as illustrated in Figure 7.4 with $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}$ representing the uncertainties around the polynomial lane marking representation.
To reduce the computational complexity of the task, the vector $\mathbf{y}_{l,r}$ is composed of a multitude of connected segments. This procedure allows dividing the sensor's research zone into Region Of Interests (ROI) on which the detection process can be performed. These zones are delimited by the uncertainties from the covariance matrix $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}$ as shown in Figure 7.4 with the blue rectangles.

Camera Data When dealing with camera data, the polynomial model has to be projected into the image space (u, v). According to [18] the projection of the road model for the left and right ego-lane marking in the image frame (u_i, v_i) is defined as follows:

$$u_i = e_u \left(\left(\frac{e_v Z_0}{(v_i - e_v \alpha)} \right)^2 \frac{c'}{6} - \frac{e_v Z_0}{2(v_i - e_v \alpha)} c_0 + \frac{v_i - e_v \alpha}{e_v Z_0} \left(y_0 \pm \frac{L_w}{2} \right) - \psi \right) \tag{7.8}$$

where $e_u = f/du, e_v = f/dv$, f is the focal distance of the camera, du and dv are the width and height of a pixel in the image, Z_0 is the height of the camera, y_0 the lateral distance of the ego-vehicle with respect to the ego-lane marking, the $\pm$ sign indicates whether the ego-marking is right $(+)$ or left $(-)$, α is the camera tilt angle and L_w is the road width. $i = 1, ..., n_{Roi}$ with n_{Roi} the number of Region Of Interest (ROI) for each lane marking.
Using equation 7.8, we can express the probabilistic model $(\mathbf{x}_{l,r}, \mathbf{\Sigma}_{\mathbf{x}_{l,r}})$ into the image space. This model will be note as $(\mathbf{u}', \mathbf{\Sigma}_{\mathbf{u}'})$, with $\mathbf{u}'$ being the average values for the pixel in the image and $\mathbf{\Sigma}_{\mathbf{u}'}$ its corresponding covariance matrix.

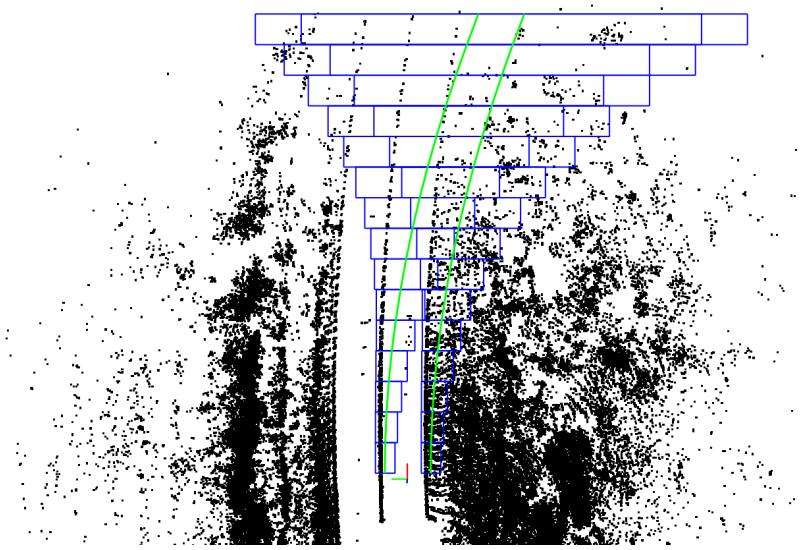


Figure 7.4: Intensity image of the lidar points with the model initialization of the presented ego-lane marking model. The green line is used for the mean value of the probabilistic model represented by the vectors $\mathbf{y}_{l,r}$, and the blue rectangles are used to represent the uncertainty $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}$.

Hence, to project this probabilistic model into the camera space, we follow the same strategy adopted for the lidar data and discussed in the section. Accordingly, the Jacobian matrix related to this projection using the Equation 7.8:

$$J_u = \begin{bmatrix} \frac{\partial u_1}{\partial c'} & \cdot & \frac{\partial u_1}{\partial x_0} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \frac{\partial u_n}{\partial c'} & \cdot & \frac{\partial u_n}{\partial x_0} \end{bmatrix} \tag{7.9}$$

As consequence, the covariance matrix $\mathbf{\Sigma}_{\mathbf{u}'}$ is calculated as follows:

$$\mathbf{\Sigma}_{\mathbf{u}'} = J_u \Sigma_{\mathbf{x}_{l,r}} J_u^T \tag{7.10}$$

The resulting projection of the Equation 7.8 is shown in Figure 7.5. Taking into account the prior about the road geometry allows focalizing the zones of research in a Top-Down process fashion. Indeed, not all the image is used to perform the detection process. Consequently, the recognition process is faster, and subject to less noise considering it takes into account only the regions in the image that most likely contains a lane marking.

7.3.2/ SELECTION OF THE BEST REGION OF INTEREST (ROI)

Once the probabilistic model is defined, we proceed to the selection of the most informative ROI in the sensor space in a **Top-Down process**. This selection strategy answers the following questions: what is the ROI in which the algorithm will have the best probability to detect a pattern (a piece of the lane marking), and at the same time, what is the ROI that will reduce by the greatest value the model uncertainties ?

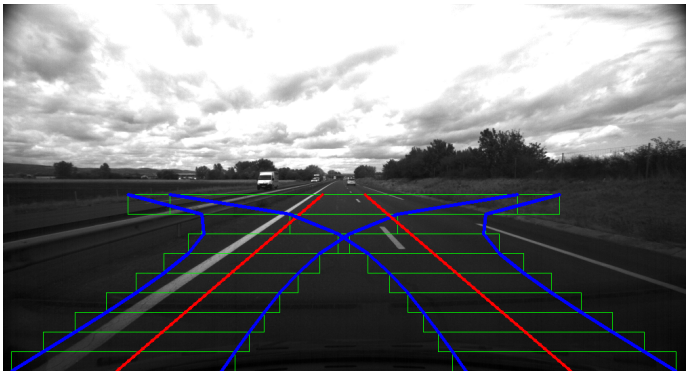


Figure 7.5: Projection of the initial probabilistic model $(\mathbf{u}', \mathbf{\Sigma}_{\mathbf{u}'})$ on the image. The mean values of the lane marking $\mathbf{u}'$ are presented in red, blue illustrates the dispersion around these values (1 σ). Finally, green boxes are the ROI for each lane marking, in this example the number of ROI is 9 for each lane marking.

To respond to these questions, we defined a unified criterion that is responsible for choosing the most suitable ROI in an *a priori* way. In our work, this criterion is based on the information theory and particularly on the Shannon entropy. Indeed, there is a strong correlation between entropy and uncertainty. The entropy is inversely proportional to the size of the quantity of uncertainties. Another way to look at it is to consider the size of the co-variance matrix. This means that a successful detection in the most informative ROI will lead to the smallest covariance matrix among the other ROI, for example; choosing the largest ROI. Nevertheless, choosing the most informative ROI in terms of precision (i.e, reducing the uncertainties) is not sufficient. Indeed, we have to make sure that the detection in this ROI has the highest probability of success. If we choose the largest ROI it will potentially cause false detections, which are due to the presences of outliers. Therefore, we must look to the probability to detect the right pattern (segment) in this chosen ROI.
As consequence, we adopted an *a priori* selection strategy based on an entropic criterion H_{sel} inspired by the work of Delobel *et al.* [125]. The idea is to choose the ROI that will maximize the entropy gain between the current' state entropy noted as H_v^-, and the entropy when we simulate a successful detection (and update) H_v^*. Hence, for each ROI we will simulate a successful detection, and we will calculate the entropic selection criterion H_{sel} as follows:

$$\begin{aligned} H_{sel} &\triangleq H_v^- - H_v^* \\ &= P(D_k = 1, D_b = 1).\frac{1}{2}\left[log|2\pi\, e\, \mathbf{\Sigma}_{\mathbf{x}}^-| - log|2\pi\, e\, \mathbf{\Sigma}_{\mathbf{x}}^*|\right] \end{aligned} \tag{7.11}$$

with:

- $\mathbf{x}^-$ The state vector before simulation of the detection,
- $\mathbf{\Sigma}_{\mathbf{x}}^-$ Covariance matrix before simulation of the detection,
- $\mathbf{x}^*$ The state vector after simulation of the detection,
- $\mathbf{\Sigma}_{\mathbf{x}}^*$ Covariance matrix after simulation of the detection.
- $P(D_k)$ The probability to detect a primitive (a segment) in the ROI,
- $P(D_b)$ The probability to detect the right pattern (segment) in the ROI,

Therefore, the ROI with the highest value of H_{sel} will be chosen. Considering this entropic criterion, it is composed of two parts. The first part $P(D_b), P(D_k)$ is responsible for the selection of the ROI on which the detection has more probability to succeed. On the other hand, the second part $\frac{1}{2}[log|2\pi\, e\, \Sigma_{\mathbf{x}}^{-}| - log|2\pi\, e\, \mathbf{\Sigma}_{\mathbf{x}}^{*}]$ is responsible for choosing the ROI that will reduce the uncertainty of the model.
For the probabilities, $P(D_b), P(D_k)$, a BN is introduced. The presented nodes are an adaptation of the BN presented in [84]. The paradigm used in order to construct this network is therefore the same, and we invite the reader to refer to the thesis of Aynaud [94] on which a detailed section is given on how to construct such a network.
The scheme of the network is illustrated in Figure 7.6. Events modeled in the BN are the following:

- X_{k_-}, the confidence before the detection is attempted
- Z_0, the chosen landmark (the white strip) is observable in the ROI
- D_k, a landmark is detected in the focal zone,
- D_b, the correct landmark has been detected (which manages landmark ambiguity)
- X_{k_+}, the confidence after the detection is attempted (success or failure)

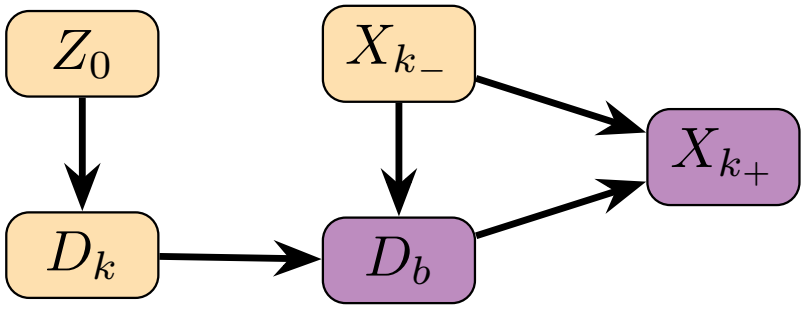


Figure 7.6: Bayesian Network used for the confidence estimation. The yellow nodes are the input nodes, the purple nodes are the nodes we are seeking to estimate.

This BN can take into account several uncertainties: the detector's performances in the node D_k, the observability of the segment in the ROI modeled in the node Z_0, the ambiguity in detecting the right segment (the one that belong to a lane marking) is modeled in the node D_b. Besides, this network has two main uses, the first one is to calculate the *a priori* probabilities $P(D_k)$ and $P(D_b)$ for the entropic criterion H_{sel}.The second one is to determine the confidence $P(X_{k_+})$ obtained after proceeding to a detection. This probability will be kept updated in this network across all the detection processes, and it will be used as **an integrity monitoring metric** that will ensure that the global model and the detected segments are coherent to the road's shape. Further application of this probability will be discussed in the end-stage. Finally, to avoid encumbering this part, all the computation related to the BN are presented in Annex 10.2.
The high-level representation in the architecture of the proposed BN allows the abstraction to the type of data sensors used. Hence, the same BN architecture is used for both lidar and camera sensors.

7.3.3/ DETECTION

Once the most informative ROI is chosen, we proceed to the detection process. The objective of this procedure is to find a segment corresponding to a part of the ego-lane marking in this chosen ROI.
When dealing with lidar data, the points cloud that are stored in the ROI are considered to be a part of the ego-lane marking. Therefore, a segment is fitted on those points. On the other hand, when dealing with the camera image we have to find the patterns that are considered as part of the ego-lane marking. As discussed in the state-of-the-art part, there exist several filters that have been introduced to detect these kinds of primitives. In our work, the implemented method is based on a row filter. The idea is to compute the gradient of the image row by row with the aim to find pixels that correspond to the lane marking. Once the patterns have been selected for the complete ROI, a fitting procedure has to be made to fit a segment on the collected patterns.
In our work, we use an extended version of the Ransac method [18]. The strategy is to detect a segment in the corresponding zone.The selected segment must satisfy two requirements in terms of the position of the middle of the segment with respect to the ROI, and the slope of the segment. If the number of patterns extracted is not sufficient, or if the selected segment does not satisfy these requirements, the detection process is considered as failure and hence the algorithm returns to the selection stage and selects the second most informative ROI. Meanwhile, the confidence in the model $P(X_k+)$ is updated by inferring the node D_k (failure) in the BN.

On the other hand, if the selected segment satisfies the requirements, two updates are performed. First, the confidence in the model $P(X_k+)$ is updated by inferring the node D_k (successful) in the BN. Second, an update of the probabilistic model is performed. Thus, for each ROI two points $p_1(u_1, v_1)$ and $p_2(u_2, v_2)$ that define the selected segment are defined. For these two points, a state vector $\mathbf{x}_p = (u_1, u_2)^\top$ is defined, we associate covariance matrix $\mathbf{\Sigma}_{\mathbf{x}_p}$:

$$\mathbf{\Sigma}_{\mathbf{x}_p} = \begin{pmatrix} \sigma_{u_1}^2 & 0 \\ 0 & \sigma_{u_2}^2 \end{pmatrix} \tag{7.12}$$

With $\sigma_{u_u}^2$representing the error in the detected segment.
In order to update the probabilistic model, we use a Kalman filter [18] as follows:

$$\begin{cases} \mathbf{u}^+ = \mathbf{u}^- + K_u \left[\hat{x}_p - x_p \right] \\ \mathbf{\Sigma}_{\mathbf{u}}^+ = \mathbf{\Sigma}_{\mathbf{u}}^- - \mathbf{K}_u \mathbf{C}_u \mathbf{\Sigma}_{\mathbf{x_p}} \end{cases} \tag{7.13}$$

$\mathbf{K}_u$ being the Kalman Gain, $(\mathbf{u}^-, \mathbf{\Sigma}_{\mathbf{u}}^-)$ is the model before detection and $(\mathbf{u}^+, \mathbf{\Sigma}_{\mathbf{u}}^+)$ is the model after detection, $\mathbf{C}_u$ is the matrix linking the vector $\mathbf{x}_p$ to the vector $\mathbf{u}'$. Nevertheless, these equations hold for both lidar and camera, only notations will change ($\mathbf{\Sigma}_{\mathbf{u}}^+$and $\mathbf{\Sigma}_{\mathbf{u}}^-$ will be replaced by $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}^+$ and $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}^-$ respectively).

7.3.4/ END

After each successful detection and update; we want to know if the ego-lane detection recognition is achieved. To perform this operation we rely on the features of the entropy

as presented in Subsection 7.3.2.
The idea is to know if enough information has been gained in comparison with the initial configuration. In other words, we compute the entropy improvement between the initial stage (the initial covariance matrix) and the actual stage (the actual covariance matrix after a successful update). However, taking into account only the entropy improvement is not sufficient. Indeed, we have to ensure that the global model used and the corresponding detected segment are coherent. To verify this coherence we rely on the probability calculated in the Bayesian network $P\left(X_{k_+}\right)$. Consequently, we define an entropic criterion H_{gain} that gather the entropy improvement between the initial covariance state $\mathbf{\Sigma_{u'}}$ and the current covariance state $\mathbf{\Sigma_{u^+}}$, and takes into account the confidence in the current state expressed by the probability $P(X_{k+})$. Thus, for the camera data, H_{gain} is computed as follows

$$H_{gain} = P\left(X_{k_+}\right).\frac{1}{2}\left[log|2\pi e\mathbf{\Sigma}^+_{\mathbf{u}}| - log|2\pi e\mathbf{\Sigma}^{'}_{\mathbf{u}}|\right], \tag{7.14}$$

We would like to emphasize that the definition of the entropic criterion H_{gain} remains the same when dealing with lidar pointcloud, only the covariance matrix $\mathbf{\Sigma_{u'}}$and $\mathbf{\Sigma_{u^+}}$ will be replaced by $\mathbf{\Sigma_{y'_{l,r}}}$ and $\mathbf{\Sigma_{y^+_{l,r}}}$ respectively.
Now, this entropy improvement is calculated, we want to compare it to a certain value to know if the recognition process is finished. In our work, we compare this criterion to the maximum entropic criterion that can be achieved. Put it another way, the entropy improvement obtained if successful detections have been in all the ROI. We define this quantity as H_{max}. Under the light of these considerations, the ego-lane marking has been detected if the following condition is satisfied :

$$H_{gain} \geqslant \lambda H_{max}, \tag{7.15}$$

with λ a fixed coefficient (< 1). If the condition is satisfied, it means that enough detections have been attempted successfully, and these successful detections have lead to a new probabilistic model which is coherent to the initialized model. In addition, it also means that the current probabilistic model is sufficiently precise to consider the ego-lane recognition finished.
If this condition is not satisfied, the ego-lane recognition process is repeated. Hence, the algorithm goes to the selection stage and chooses the second most informative ROI is selected. On the other hand, if a detection process has been tested on all the ROI remaining and the condition is still not satisfied, then it means that the current probabilistic model is not coherent and does not cover the lane marking. In these cases, the recursive nature of the algorithm allows it to go backward and backs the previous probabilistic model, which means the one before the update.
Finally, if the condition is satisfied then the ego-lane marking detection is completed and hence the recognition process is finished, it also indicates the end of the ego-lane level localization. Indeed, the update of the probabilistic model in the sensor frame leads to the update of the road model defined by $(\mathbf{x}_{l,r}, \mathbf{\Sigma}_{\mathbf{x}_{l,r}})$. Consequently, an estimation of all the road parameters is performed, and hence the lateral position y_0 of the ego-vehicle with respect to the ego-lane is computed.

7.3.5/ TRACKING STEP

Once the ego-lane recognition process is finished, a tracking procedure is performed. The objective of this process is to provide a smaller confidence interval for the next frame

$(k+1)$ than the one provided by the initial probabilistic model (cf. subsection 7.3.1). This implies smaller values for the covariance matrix $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}$ for lidar, and $\mathbf{\Sigma}_{\mathbf{u}}$ for camera image.As a consequence, it also reduces the required computation needed for recognition frame. To do so, we define a general vector $\mathbf{x}_{all}$. When dealing with lidar data, this vector which contains the 2D vector $\mathbf{y}_{l,r}$ and the road's parameter vector $\mathbf{x}_{l,r}$ defined as follows $\mathbf{x}_{all} = (\mathbf{x}_{l,r}, \mathbf{y}_{l,r})^\top$. On the other hand, when dealing with camera images, the vector $\mathbf{x}_{all}$ contains the pixel polynomial representation $\mathbf{u}$ and the road's parameter vector $\mathbf{x}_{l,r}$. In the same logic its covariance matrix $\mathbf{\Sigma}_{\mathbf{x}_{all}}$ is defined. The initial value of this newly defined probabilistic model $(\mathbf{x}_{all}(0), \mathbf{\Sigma}_{\mathbf{x}_{all}}(0))$ can be easily computed using the same strategy presented in Subsection 7.3.1. Thus, we start computing the evolution of the model $(\mathbf{x}_{l,r}, \mathbf{\Sigma}_{\mathbf{x}_{l,r}})$ as follows[18]:

$$\begin{cases} \mathbf{x}_{l,r}\,(k+1) \;\; = \mathbf{M}\;\mathbf{x}_{l,r}(k) \;\; + \mathbf{W}_t \\ \mathbf{\Sigma}_{\mathbf{x}_{l,r}}(k+1) = \mathbf{M}\;\mathbf{\Sigma}_{\mathbf{x}_{l,r}}(k)\;\mathbf{M}^T + \mathbf{Q} \end{cases} \tag{7.16}$$

This equation is linear and holds for small time interval. Hence, $\mathbf{M}$ is the evolution matrix that includes the displacement and the angle difference. The matrix $\mathbf{Q}$ represents the error in the evolution matrix. Thereafter, we update the new model $(\mathbf{x}_{all}(k+1), \mathbf{\Sigma}_{\mathbf{x}_{all}}(k+1))$ using the model $(\mathbf{y}_{l,r}(k), \mathbf{\Sigma}_{\mathbf{y}_{l,r}}(k))$ that we get from the recognition step (for the lidar data)

$$\begin{cases} \mathbf{x}_{all}(k+1) = \mathbf{x}_{all}(0) + \mathbf{K}_t\left[\mathbf{x}_{l,r}(k+1) - \mathbf{x}_{l,r}(0)\right] \\ \mathbf{\Sigma}_{\mathbf{x}_{all}}(k+1) = \mathbf{\Sigma}_{\mathbf{x}_{all}}(0) - \mathbf{K}_t\mathbf{C}_t\mathbf{\Sigma}_{\mathbf{x}_{all}}(0) \\ \qquad\qquad \mathbf{K}_t = \mathbf{\Sigma}_{\mathbf{x}_{all}}(0)\mathbf{C}_t^{\mathrm{T}}\left[\mathbf{C}_t\mathbf{\Sigma}_{\mathbf{x}_{all}}(0)\mathbf{C}_t^{\mathrm{T}} + \mathbf{\Sigma}_{\mathbf{x}_{l,r}}(k+1)\right]^{-1} \\ \qquad\qquad \mathbf{x}_{l,r} = \mathbf{C}_t\mathbf{x}_{all} \end{cases} \tag{7.17}$$

With $(\mathbf{x}_{l,r}(0), \mathbf{\Sigma}_{\mathbf{x}_{l,r}}(0))$ being the initialized model from subsection 7.3.1. After this update, the vector $\mathbf{y}_{l,r}(k+1)$ and its covariance matrix $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}(k+1)$, contained in the model $(\mathbf{x}_{all}(k+1), \mathbf{\Sigma}_{\mathbf{x}_{all}}(k+1))$ define a new confidence interval for the next frame. The previous equations hold for the lidar data. However, the same equations are used when dealing with the camera image, the vector $\mathbf{u}(k)$ and the matrix $\mathbf{\Sigma}_{\mathbf{u}}(k)$ will replace the vector $(\mathbf{y}_{l,r}(k)$ and the matrix $\mathbf{\Sigma}_{\mathbf{y}_{l,r}}(k))$

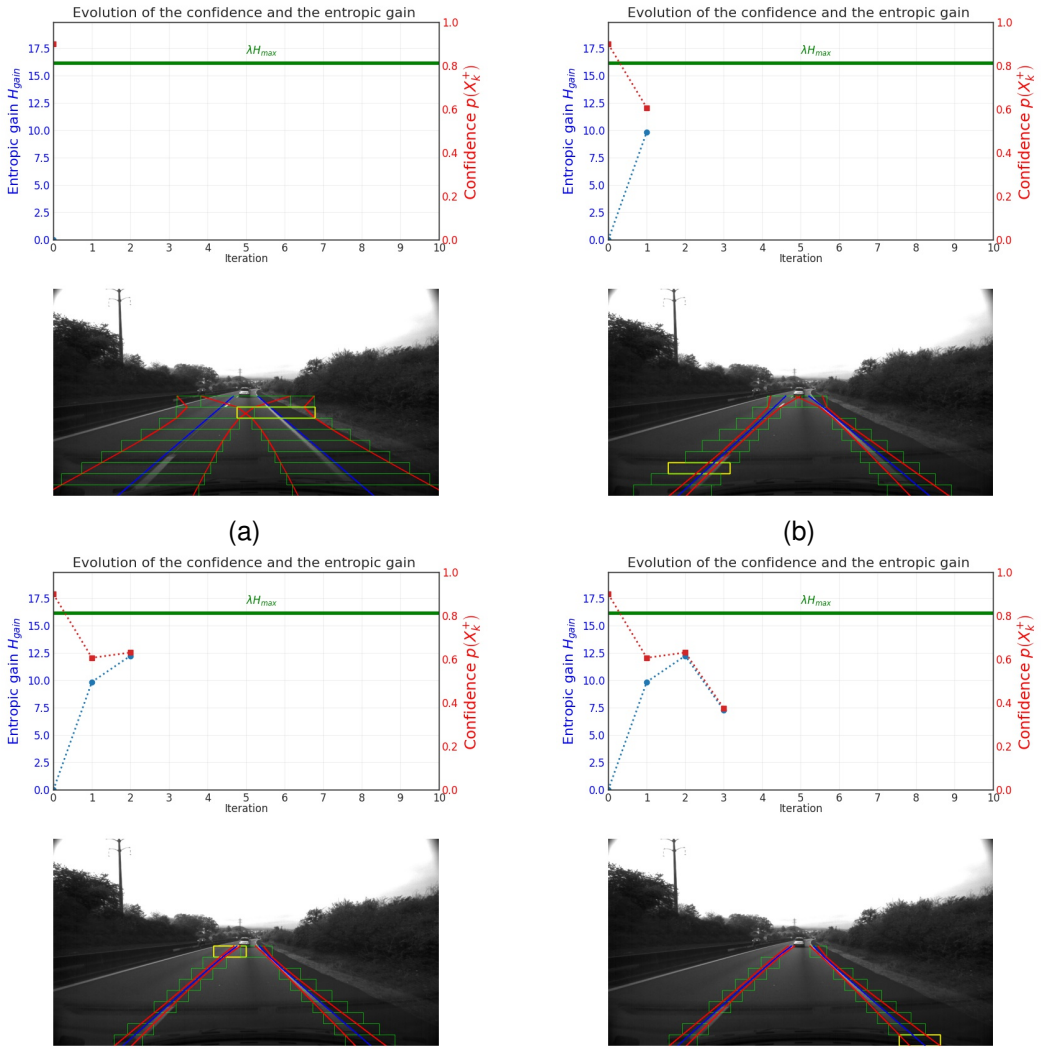
7.3.6/ CANONICAL EXAMPLE

In the following, an example of the functioning of the algorithm on an image is presented. Graphical illustration is given as a support for each step of the RIDA in Figure 7.6 in which the confidence in the model and the entropy improvement are shown at each step:

- Figure 7.6a Illustrates the initialization stage on which the probabilistic model $(\mathbf{u}', \mathbf{\Sigma}_{\mathbf{u}}')$ is projected in the image space and the initial confidence about the model is set. The mean values of the lane marking contained in the vector $\mathbf{u}'$ are shown in blue, and the uncertainties coming from the covariance matrix $\mathbf{\Sigma}_{\mathbf{u}}'$ are shown in red. Once the probabilistic model is set, the area of research are divided into ROI (in this example, 9 regions for each edge). Afterward, the most informative ROI is chosen (illustrated in yellow) and detection is attempted in this ROI
- Figure 7.6b The detection attempted was successfully leading to an update to the probabilistic model and hence reducing the ROI area. Although, even if the detection was successful, the confidence in the model goes lower. Indeed, the shrinkage

of the uncertainties' area leads to a loss of integrity. In contrast, the entropy gain goes higher. Afterward, the next most informative ROI is chosen and detection is attempted.

- Figure 7.6c to Figure 7.6i the procedure of selection, detection, and update is repeated until the entropic gain exceeds the threshold $\lambda\ H_{max}$. Once the entropic gain is superior to this threshold, the lane marking detection is considered as finished as illustrated in Figure 7.6j



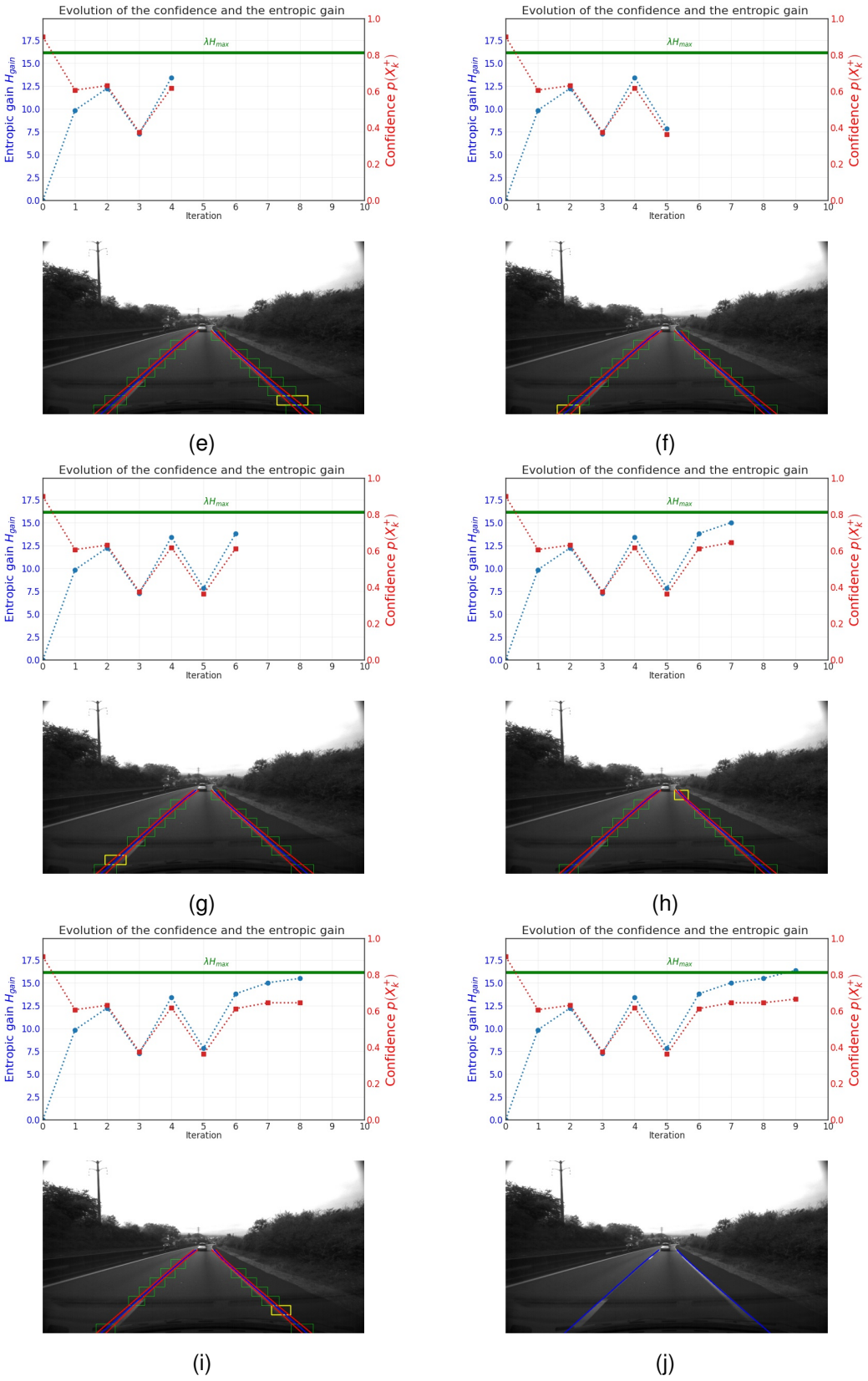


Figure 7.6: All steps for the RIDA. In the upper part of each image, red is used for the confidence about the probabilistic model, blue for the entropic gain, and the green line represents the threshold for which the recognition is considered finished. In the lower parts of each image the lane marking are illustrated in blue, the uncertainties about these marking is show in red, green is used for theROI and yellow for the selected ROI. Details about each image are given in text.

7.4/ RESULTS AND DISCUSSION

In order to show the effectiveness of our algorithm several experimental results have been conducted on both camera images and lidar data. However, as discussed in Chapter 3 the evaluation of the majority of lane marking algorithms for camera images is made on the pixel level. This type of evaluation is not suitable for our case, since we do not take into account all the pixels of the image that belong to the lane marking, but we model the curvature of the road with a polynomial curve. In addition, the metrics that are used in the literature have the downside of not capturing the global coherence of the detected lane marking, which is translated in the shape of the ego-lane. However, qualitative results in the form of videos can be found following this link:shorturl.at/ajpG7.
Under these circumstances, we tested our algorithm on lidar data.

7.4.1/ THE DATASETS

Therefore, to illustrate the effectiveness of our lidar-based lane detection system, we investigate the available datasets for autonomous vehicles. We focus our attention on the datasets suitable for lane marking detection in highway scenarios. Taking into account these considerations, it appears that most of the available datasets are not suitable to qualify the effectiveness of our algorithm, mainly for two reasons. The first one is that most of the datasets do not provide lidar data. The second one is the type of annotation for the Ground Truth. As highlighted in Chapter 3, there does not appear to be a consensus among the autonomous driving community on the type of annotation for ego-lane marking. In reality, most datasets use pixel-wise annotation (e.g, KITTI [70], Ego-Lane Analysis System (ELAS) [110], Unsupervised Llamas [141]). This pixel-level annotation is not suitable for us. Indeed, we have a higher-level representation of the lane, as we represent the ego-lane marking as a polynomial. Of course, someone may point out the Carina datasets [88]. However, to the author's knowledge, these datasets are still not available. The absence of suitable ground truth to analyze the effectiveness of our lidar-based lane detection system brought us to build our own annotation using the KITTI datasets [70].

7.4.2/ CONSTRUCTION OF THE DATASET

In order to evaluate our ELL algorithm, we have to build the ground truth from the KITTI datasets by manually annotating the lidar points to determine the ego-lane. The strategy adopted is to transform the lidar pointcloud into a Bird's Eye View image as illustrated in Figure 7.7 . This transformation allows us to get an image on which we can perform pixel annotation for the lane marking. In our experiments, the image has a dimension of 240 x 600 pixels with each pixel has a size of 0.05 x 0.05 m. This procedure is repeated for all the two highway sequences from the KITTI datasets noted as "2011_09_26/0027" and "2011_09_26/0028".
Once the BEV images available, we annotate the ground truth for the ego-lane as illustrated in Figure 7.7. Therefore, all the evaluations will be conducted in this BEV space.

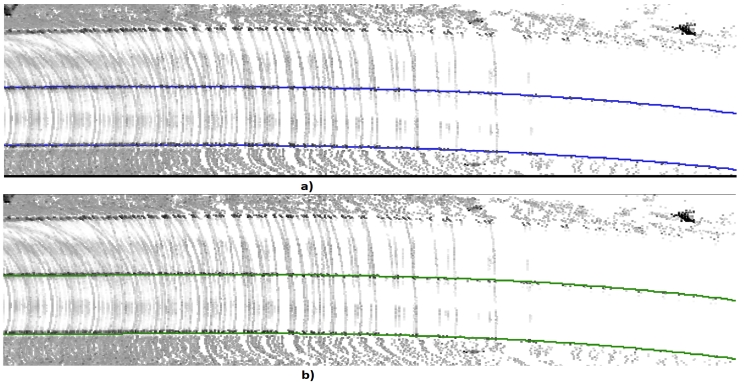


Figure 7.7: Process for the evaluation of the detection lane, first a BEV (Bird's Eye View) image is produced with its associated ground truth in blue (a), the result of the detection is transposed in the BEV image in green (b).

7.4.3/ QUANTITATIVE RESULTS

In order to quantify evaluate the effectiveness of our algorithm, statistical analyses have to be made for the detection performances. Thereby metrics have to be chosen. In that context, several metrics are used in the literature [124]. However, as it has been concluded in Chapter 3, the majority of these metrics are not suitable to be used to evaluate our algorithm. Indeed, we need a metric that enables us to distinguish between an error of 1 cm and one of 1 m. For that reason, we use Root mean squared error (RMSE). Indeed, the BEV representation allows us to have a metric representation of the ego-lane marking. Furthermore, the RMSE metric has the benefit of penalizing large errors. Nevertheless, in order to accomplish this evaluation, parameters related to our probabilistic model have to be defined. These parameters are shown in Table 7.1. The authors would like to emphasize that these parameters were set by hand and no extensive hand tuning was performed. In addition, these parameters are means values that represent global knowledge about road's shape model and are suitable for highway scenarios. The corresponding results for the RMSE are summarized in Figure 7.8. The average values for the whole sequences are presented in Table 7.2. The results presented testify about the effectiveness of our proposed algorithm. Accordingly, by taking into account the RMSE in Figure 7.8, we found that more than 97.80% of the time the detection error is below 0.15 m

variable	definition	value
n_{roi}	number of ROIs	20
μ_{x_0}	mean value for distance to center line	0 m
μ_{psi}	mean value for ψ	0 rad
μ_{L_w}	mean value for lane width	3.5 m
$\sigma_{c'}$	standard deviation for the curvature's derivative	$10^{-5}m^{-2}$
σ_{c_0}	standard deviation for the curvature	$10^{-4}m^{-1}$
σ_{x_0}	standard deviation for distance to center line	0.25 m
σ_{psi}	standard deviation for ψ	0.1 rad
σ_{L_w}	standard deviation for lane width	0.5 m

Table 7.1: Parameters used for our experiments

Metrics	Sequence 27	Sequence 28
RMSE (mean)	0.07 m	0.11 m
RMSE (median)	0.06 m	0.10 m
RMSE (max)	0.40 m	0.48 m

Table 7.2: Results of RMSE for our experiments on the KITTI dataset.

for sequence 27 and 80.61% for sequence 28. Nevertheless, in the worst cases, the maximum values in Table 7.2 are 0.40 m and 0.48 m which shows that even when incorrect detections happen, the error is still acceptable.

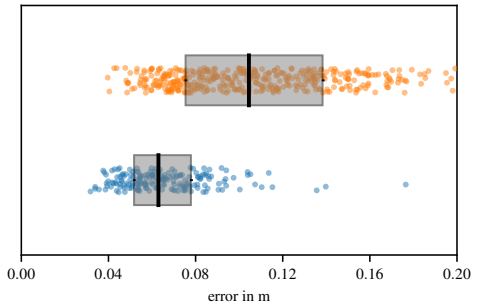


Figure 7.8: RMSE of the sequence 27 and of the sequence 28. The color code used is the follow: Orange stands for sequence 27 and Blue for sequence 28

Furthermore, when these incorrect detections occur, the outcome result is still coherent as shown in Figure 7.9. By investigating these cases we found that these scenarios happen when the density of the lidar is not enough sufficient for a good ego-lane marking detection.

7.4.4/ QUALITATIVE RESULTS

Some qualitative results concerning ego-lane detection are presented in Figure 7.9. For a visual comparison, the ego-lane marking detected was projected on the image and a camera-based detection solution was added.
The presented results show that the proposed Lidar-based solution is suitable to detect the ego-lane marking. In Figure 7.9(b) an illustration of a case where the high brightness makes the detection of the ego-lane marking difficult for the camera. However, it is not the case when using the ego-lane marking algorithm with the lidar data. The most remarkable attribute of the proposed algorithm appears when a miss-match happens, as illustrated in Figure 7.9(d). Indeed, the outcome is still coherent, the geometry of the lane is preserved. This feature is due to the probabilistic model $(\mathbf{y}_{l,r}, \mathbf{\Sigma}_{\mathbf{y}_{l,r}})$, as we take into consideration the shape of the road from OSM as prior. As a result, even if a miss-match occurs, the result of the detection is still coherent, which is not the case when using a traditional data-driven approach. In [124] when miss-matches occur, it results in a cross-shaped lane marking that is not coherent to the shape of a road, ultimately this kind of method can not be solely used for an autonomous vehicle application. Besides our solution shows good results when detecting the ego-lane marking, it guarantees that even if the detecting process

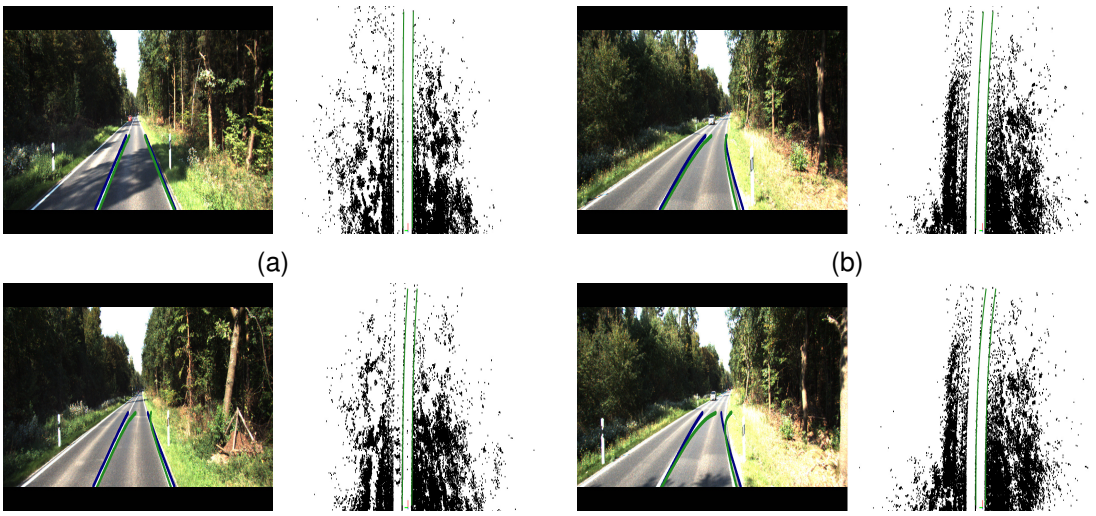


Figure 7.9: Some examples of detections using the lidar, the results were projected into the image for a visual comparison with a camera solution. (a) shows a case where the lidar and camera solution correctly detects the ego lane, (b) shows a case where the camera fails due to high brightness, (c) the lidar fails where the camera does not, in (d) both of the solution fail. The color code used is as follows: green for the mean value of the ego-lane marking using lidar and blue for camera. An additional video is provided in shorturl.at/bktvE

goes wrong, the outcome will be still coherent and acceptable. The work presented in this chapter have been published in ICARCV 2020 : 16th International Conference on Control, Automation, Robotics and Vision (ICARCV20) [165] and have been selected for the Best Student Paper (5 candidates).

7.5/ CONCLUSION

This chapter details the ELL solution proposed in this P.h.D work. The global architecture proposed allows the detection of ego-lane marking for both lidar and camera images.
The proposed framework relies on an information-driven approach and is designed to benefits from inaccurate inputs that represent priors on road's geometry from OSM. To do so, a probabilistic model was introduced in order to restrict the region of interest in the sensor domain (camera or lidar). Moreover, this probabilistic model ensures the coherence of the results of the ego-lane detection, since it takes into account the geometry restriction of the lane. Furthermore, the presented framework guarantees that the outcome of the detection is coherent to the road geometry, as the shape of the detected ego-lane marking is coherent even if miss-matches occur.
For the recognition process, we proposed a Recursive Information-Driven Algorithm (RIDA) that uses entropy features in order to select the most information area of research to reduce the space complexity. The mathematical formalizing of each block composing the RIDA have been justified. In addition to that, the robustness of the proposed method is proven on real datasets and statistical metrics are used to highlight our method's efficiency.

In the next chapter, the third part of the proposed end-to-end localization framework is introduced. It corresponds to the LLL algorithm proposed in this P.h.D work.

8

LANE LEVEL LOCALIZATION SOLUTION

Contents

8.1 Problem formulation 122
8.2 Modular Probabilistic Framework For lane level localization 123
8.2.1 Adjacent lanes extrapolation 123
8.2.2 Bayesian network for ego-lane determination 124
8.2.3 Hidden Markov Model Lane-Level Localization 127
8.2.4 Transition probability (Lane change probability) 128
8.3 Real-world experimental results 130
8.3.1 Quantitative results 130
8.3.2 Qualitative results 131
8.4 Conclusion 131

8.1/ PROBLEM FORMULATION

Knowing the position of the host lane can be utilized by the autonomous navigation system to provide the most adequate instructions. The navigation system may be required to recommend lane change to driver in order to maintain the driver's safety. Therefore, an accurate but also reliable Lane-Level Localization (LLL) is needed.
The state-of-the-art review in Chapter 4 presented the existing solutions in order to perform the LLL. In the broadest sense, LLL can be performed in two distinct ways. The first category relies on a high-definition map that contains landmarks of the road scene. The methods that fall into this category are highly tributary to the quality of the map used. In contrast, the second category relies on the detection of landmarks by on-board sensors. Contrary to the first category, these methods do not suffer from the HD map's exorbitant cost to build and maintain. Therefore, we present a Modular Probabilistic Framework for lane level localization that relies on on-board sensors, ELL and lanes number information from OSM. The novelty relies upon the probabilistic framework developed, as we introduce a modular BN to infer the ego-lane position from multiple inaccurate information sources. The flexibility of the BN is proven, by first, using only information from surrounding lane-marking detection, and second, by adding adjacent vehicles' detection

information. Afterward, we design a HMM to temporarily filter the outcome of the BN using the lane change information. The effectiveness of the algorithm is tested are validated on challenging scenarios and compared to an existing method, whose authors made their datasets public. In the following sections, details about the proposed LLL are given.

8.2/ MODULAR PROBABILISTIC FRAMEWORK FOR LANE LEVEL LOCALIZATION

Once the ego-lane localization is estimated, we have to perform the lane-level localization in order to correctly choose the right lane on which the vehicle travels. To do so, we propose a probabilistic framework that is split into three stages.
In the first stage, we extrapolate adjacent lanes by assuming that lanes in the same road have the same width L_w and same curvature's parameters c', c_0. The second step consists of a Bayesian Network (BN), that takes as input the results of the hypothesized adjacent lane-marking detection, whether these detections succeed or fail in order to determine the ego-lane localization. Furthermore, since the proposed BN is modular, we also use an adjacent vehicle detector that will serve to solve some ambiguous situations. The third step includes a filtering process, using a Hidden Markov Model (HMM) in order to take into account physical constraints of the ego-vehicle.

8.2.1/ ADJACENT LANES EXTRAPOLATION

The adjacent lanes are extrapolated by taking advantage of the estimated ego-lane localization and the number of lanes from OSM. Thus, each adjacent lane is described by a probabilistic model $\{\mathbf{x}_a, \mathbf{\Sigma}_{\mathbf{x}_a}\}$, where $\mathbf{x}_a$ contains the parameters $(c', c_0, \psi, L_w, y_0)^T$ described in Chapter 7 and $\mathbf{\Sigma}_{\mathbf{x}_a}$ refers to the corresponding covariance matrix. The assumption made is that the curvature and lane width stay constant for all the lanes in the same road. Thereby, the value of the vector $\mathbf{x}_l$ remains the same as $\mathbf{x}_{l,r}$, only the value of y_0 will be shifted by a $(\pm iL_w)$ with $(-i)$ indicates that the edge is on the right of the ego-lane and $(+i)$ on the left. The number of adjacent lanes extrapolated is equal to the lanes number, which is assumed to be known from OSM (cf. Chapter 6).
Thus, from a perspective view, the hypothesized lanes are shown in Figure 8.1.

Following the same strategy adopted in Chapter 7, the probabilistic model $\{\mathbf{x}_a, \mathbf{\Sigma}_{\mathbf{x}_a}\}$ can be transferred to the model image $\{\mathbf{u}_a, \mathbf{\Sigma}_{\mathbf{u}_a}\}$. As a consequence, $\mathbf{u}_a$ represents the horizontal pixel of the edges in the image and $\mathbf{\Sigma}_{\mathbf{u}_a}$ its interval confidence. In Figure 8.2, the adjacent lane regions of interest resulting from the extrapolation are shown on the image. In addition to focusing the detection on a specific region of the image, this procedure allows us to keep the global coherence between the adjacent lane marking and the ego-lane marking detected before.

For each adjacent lane marking, a detection process is performed. The results of this detection, or in other words, the Boolean that indicates if the detection has been successful or not, is fed into a designed Bayesian Network (BN). The events modeled in this network, as well as the network's objective, are described in more detail below.

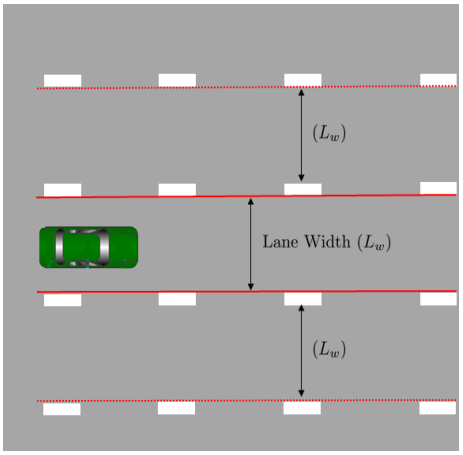


Figure 8.1: Perspective view, ego-lane in solid lines and hypothesized edges in dotted lines (case three lanes road)

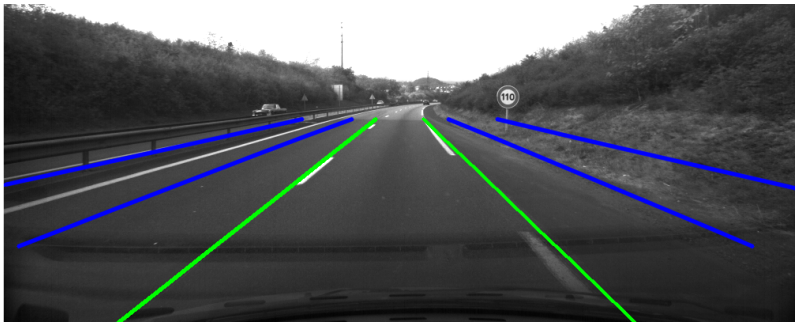


Figure 8.2: Confidence intervals for adjacent hypothesized edges for a two lanes road. Green represents the estimated edges of the ego-lane and blue illustrates the confidence intervals for the possible positions of the adjacent lanes marking.

8.2.2/ BAYESIAN NETWORK FOR EGO-LANE DETERMINATION

The BN proposed is designed to be flexible and modular for other detection results that may arrive from any type of sensors, *i.e.* vehicle detector, guardrail detector. As a consequence, its architecture is similar to the one presented by Nieto *et al.*[47]. The idea to have a semi-fixed BN in which the addition of other detection results will not lead to the reshaping of the network. Towards this end, a strong assumption is made. Indeed, in this network, the detection results from the different detectors are considered independent. Although this assumption is not always guaranteed. Indeed, some detections can be correlated (i.e., the non-detection of a lane marking can be caused by the presence of a vehicle). However, it allows the modality and flexibility of the proposed BN. As consequence, the general architecture of the BN used is illustrated in Fig 8.3. The nodes described are the following:
- Zk_i: The element i is observable, this element can represent any element of the road scene: {vehicles, lane-marking, traffic signs...},
- Dk_i: The detection of the element i is successful,
- L_{bn}: The lane on which the ego-vehicle is traveling $L_{bn} = \{l_1, ..., l_n\}$, where l_1 indicates

the leftmost lane and n the lanes number.
A deep look into this architecture, there exists similarities between this BN and the Bayes naive classifier architecture that are used for classification problematic. The likeness does not limit itself on the graphical representation. In the Bayes naive classifier, the nodes are considered as independent. However, the Bayes naive classifier is considered one of the most powerful classifiers in the machine learning domain. For those reasons, we think that this assumption will not hurt the accuracy provided by our solution. Furthermore, in

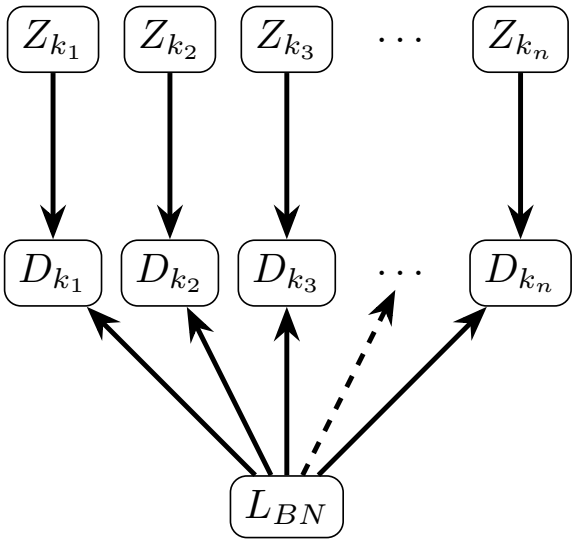


Figure 8.3: Graphic representation of the general architecture of the Bayesian Network used for the ego-lane determination

order to show the flexibility of the BN, we will first use only adjacent lanes' detection to determine the ego-lane, then, we will introduce an adjacent vehicle detector based on Deep Learning solution available in the literature (YOLO [104]).
Depending on the results of the detection, we infer the probability to belong to a lane $\{l_1, l_2, ..., l_n\}$ as follows:

$$p(L_{bn=l_i}) = p(L_{bn=l_i}|Dk_1, Zk_1, ..., Dk_n, Zk_n) \tag{8.1}$$

8.2.2.1/ BAYESIAN NETWORK USING ONLY ADJACENT LANES DETECTION (BN+ALD)

To start, we only use the adjacent lanes detections as input into our BN. Thus, the ego-lane determination will be determined considering the result of these detections, whether these succeed or fail. Figure 8.4 shows results of the corresponding BN. As illustrated, there are some cases where the BN is unable to overcome the ambiguity to locate the ego-lane. Indeed, this happens when the adjacent edges are not detected or wrongly detected.

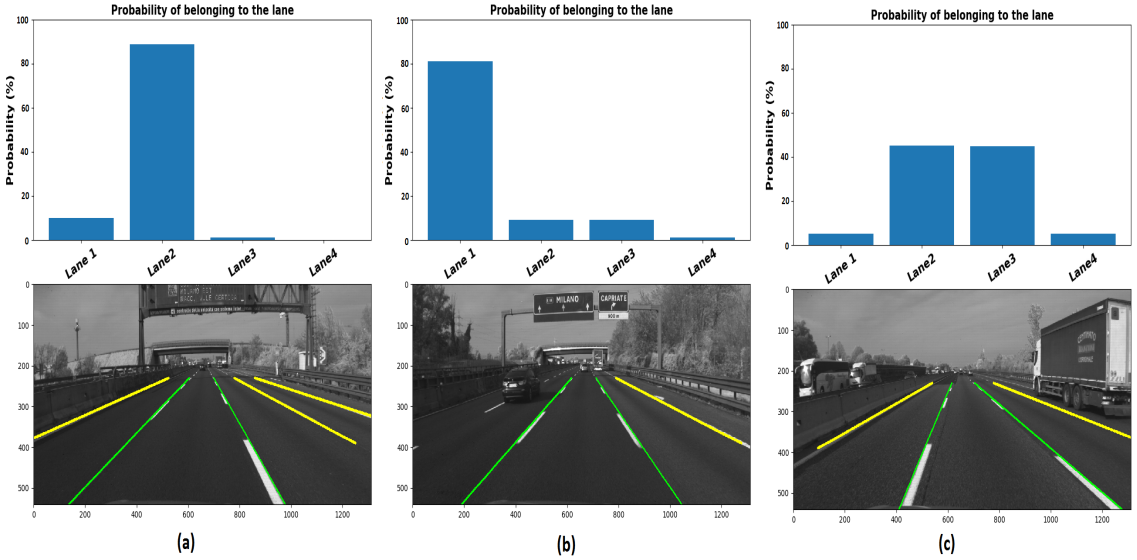


Figure 8.4: Results of ego-lane determination using the BN with only the adjacent lanes' detection. In (a) the BN success as the marking are correctly detected, in (b) and (c) the adjacent lanes are not detected, which leads to a false result. In all figures, green represents ego-lane marking and yellow represents adjacent lanes marking detected

8.2.2.2/ BAYESIAN NETWORK WITH THE ADDITION OF THE ADJACENT VEHICLE DETECTOR (BN+ALD+AVD)

As mentioned before, the BN proposed is aimed to be modular for other detectors. Thus, we use the same BN architecture presented earlier and add the information about the adjacent vehicles' detection.
The final Bayesian Network used is illustrated in Figure 8.5, where Dl_i indicates whether the detection of adjacent lane i is successful. Concerning Dv_j, it indicates whether the detection of the adjacent vehicle j is successful. Knowing that this detection is performed in the regions of the image bounded by the neighboring marking lanes.

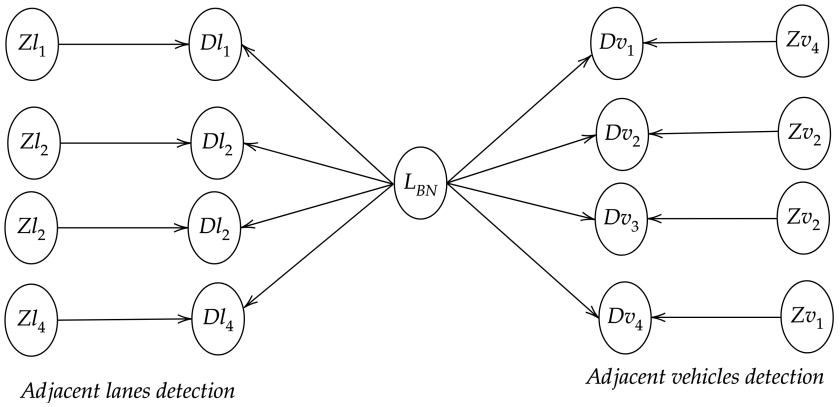


Figure 8.5: BN with adjacent lanes and vehicles detection (case 4 lanes road)

As a result of this addition, the determination of the ego-lane is affected as shown on

Figure 8.6.

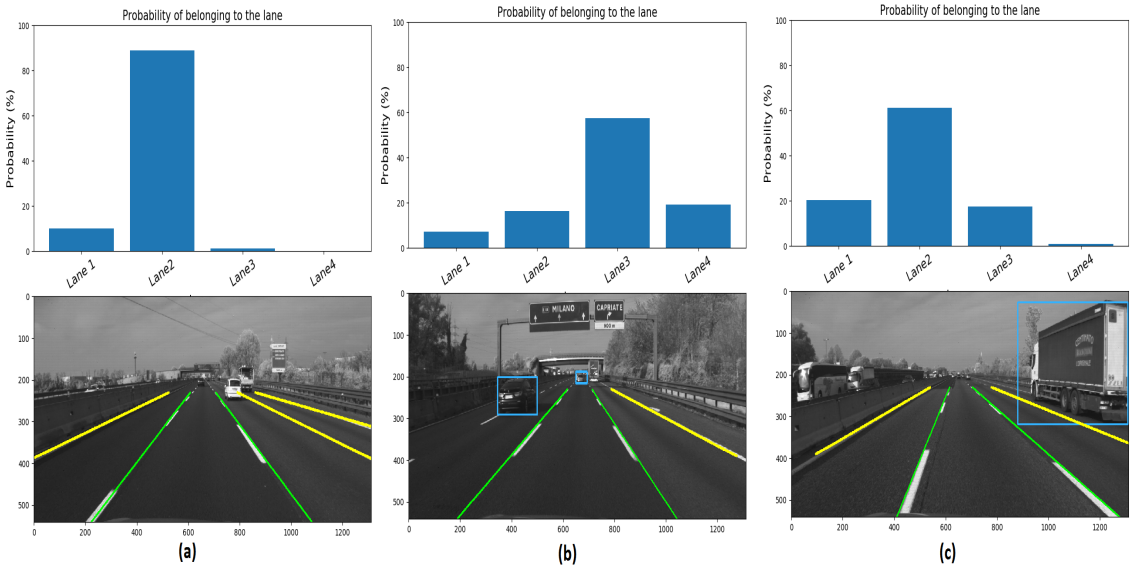


Figure 8.6: Results of ego-lane determination using the BN with the adjacent lanes and vehicle detection. For the case (a), the addition of the vehicle detector does not change the result, since no vehicle is detected. However, in (b) and (c) the vehicle detector leads to a correct determination of the ego-lane. In all figures, cyan indicates the bounding box of vehicles detected on the image.

8.2.3/ HIDDEN MARKOV MODEL LANE-LEVEL LOCALIZATION

The BN proposed in the section before is applied on a per-frame basis. However, the dynamic relationship between two consecutive frames is not taken into account. Indeed, the BN does not take into consideration the dynamic constraints of the ego-vehicle (*i.e*, the ego-vehicle can only change lane to an adjacent lane). In order to take into account these constraints, we filter the output of the BN by a Dynamic Bayesian Network which is the Hidden Markov Model (HMM).
We will use L_t to denote the set of ego-lane state variables at time t, which depends on the lanes number n_{lanes} and which are assumed to be observable. e_t denotes the observable evidence variable. The aim of the filter algorithm is to estimate the probability $P(L_{t+1}|e_{1:t+1})$. According to [66], this probability can be formulated as follows:

$$P(L_{t+1}|e_{1:t+1}) = P(L_{t+1}|e_{1:t}, e_{t+1}) \text{ (dividing up the evidence)} \tag{8.2}$$
$$= \alpha P(e_{t+1}|L_{t+1}, e_{1:t}) P(L_{t+1}|e_{t+1}) \text{ (Baye's rule)} \tag{8.3}$$
$$= \alpha P(e_{t+1}|L_{t+1}) P(L_{t+1}|e_{t+1}) \text{ (Markov assumption)} \tag{8.4}$$

Where α denotes a normalizing constant to make the sum of probabilities equal to 1. By arranging Equation 8.4:

$$P(L_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|L_{t+1}) \sum_{l_t} P(L_{t+1}|L_t, e_{1:t}) P(L_t, e_{1:t})$$
$$= \alpha P(e_{t+1}|L_{t+1}) \sum_{l_t} P(L_{t+1}|L_t) P(L_t, e_{1:t}) \tag{8.5}$$

The probability $P(e_{t+1}|L_{t+1})$ comes from the observation model, hence in this work from the BN described previously, thus:

$$P(e_{t+1}|L_{t+1}) = P(L_{bn}) \tag{8.6}$$

With regard to the probability $P(L_{t+1}|l_t)$, it comes from the transition model. It expresses the probability of the ego-lane to change its current state, which is the lane change probability.

Finally, the third term $P(L_t, e_{1:t})$ expresses the current state distribution. Graphically, we can illustrate the corresponding HMM as in Figure 8.7.

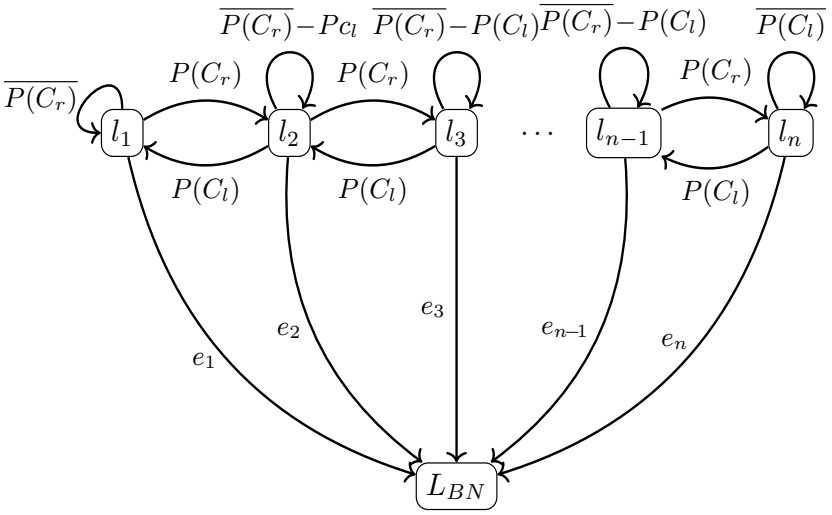


Figure 8.7: Hidden Markov Model for n lanes case, with $L = \{l_1, l_2, ..., l_n\}$ being the set of hidden states and $O = \{e_1, e_2, ..., e_n\}$ being the set of observations resulting from the BN.

With the recursive formulation obtained in Equation 8.5, we can estimate the current ego-lane state given the observation obtained from the BN. But before, we have to compute the lane change probability.

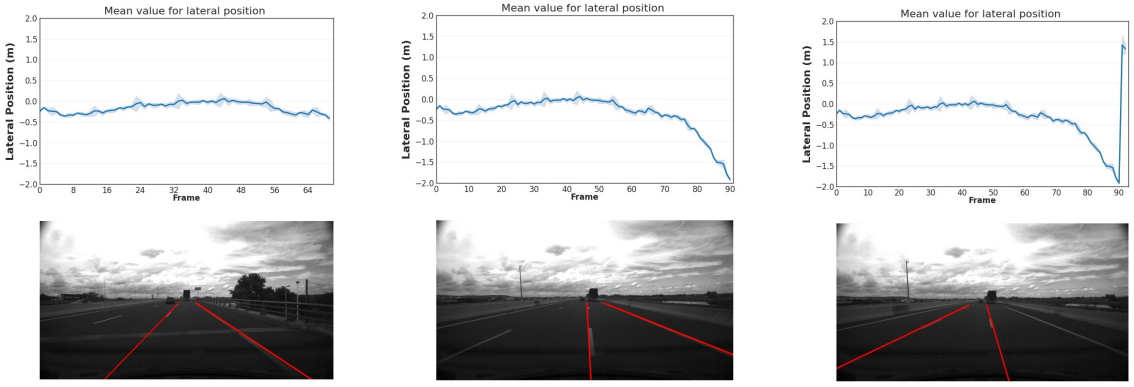
8.2.4/ TRANSITION PROBABILITY (LANE CHANGE PROBABILITY)

To calculate the lane change probability, we model the lateral position y_0 estimated in Chapter 7 as normal distribution with mean μ_{y_0} and variance $\sigma^2_{y_0}$:

$$y_0 \sim \mathcal{N}(\mu_{y_0}, \sigma^2_{y_0}) \tag{8.7}$$

The value μ_{y_0} and variance $\sigma^2_{y_0}$ are obtained from the estimated probabilistic model calculated by the ELL algorithm (cf. Chapter 7), as illustrated in Figure 8.8.

Starting from this estimation, we predict the lateral position y_0 at t_{k+1} using the proprioceptive information of the ego-vehicle. Accordingly, the lane-change probability as shown



(a) (b) (c)

Figure 8.8: Estimation of the lateral shift μ_{y_0} and its variance σ_{y_0} (3 σ) in case of lane change.

on Figure 8.9, is calculated as follows:

$$P(C_r) = \int_{L_w/2}^{\infty} \frac{1}{\sigma_{y_0}\sqrt{2\pi}} e^{-\frac{(x-\mu_{y_0})^2}{2\sigma_{y_0}^2}} dx \tag{8.8}$$

$$P(C_l) = \int_{-\infty}^{-Lw/2} \frac{1}{\sigma_{y_0}\sqrt{2\pi}} e^{-\frac{(x-\mu_{y_0})^2}{2\sigma_{y_0}^2}} dx \tag{8.9}$$

With $P(C_r)$ the probability to right change lane and $P(C_l)$ the probability to left change lane.

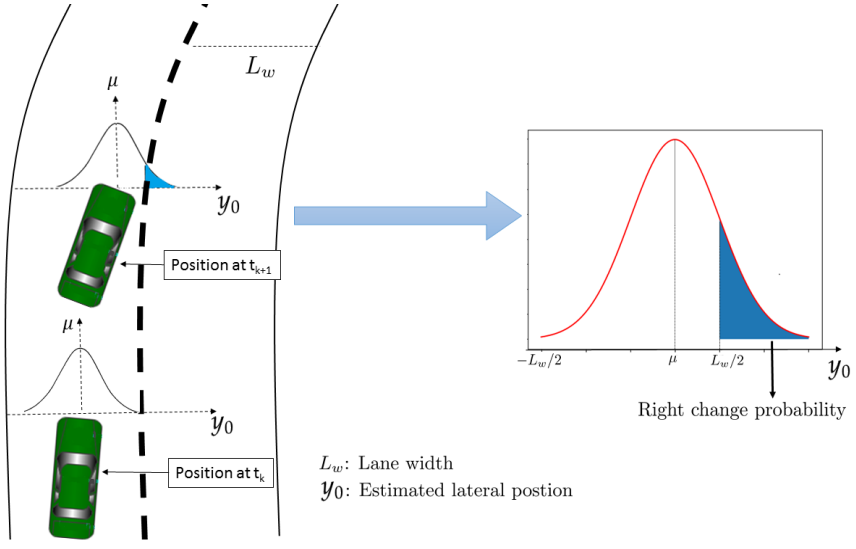


Figure 8.9: Lane change probability, with the blue area representing the right change probability.

Now, that we have designed the HMM, we will introduce the real-world experimental results on the following section.

8.3/ REAL-WORLD EXPERIMENTAL RESULTS

In order to evaluate our proposed method, we tested our algorithm on real driving data sets. The first one was collected in the region of Clermont-Ferrand in France, where we drove our vehicle on a two lanes road on the national highway for a total of 838 frames. Given the number of frames, these datasets were only used to show the effectiveness of the algorithm. In addition, we wanted to compare the performances of our algorithm to the literature on more challenging scenarios. Naturally, we turned our attention to the KITTI datasets [77]. However, these datasets contain only few lanes and few lane-changing scenarios. Thus, we tested our algorithm on some datasets refereed in [109] [1]. Unfortunately, all the data were not ready yet. But, we were able to test on some of them, noted as the A4-Highway Italy, for a total of 9528 frames. The collected datasets were manually annotated in a per-frame basis in order to determine the correct ego-lane classification. In the following, we will refer to our datasets as "2-lanes" and the A4-Highway Italy as "4-lanes".

To show the increment of each component added, we first determined the ego-lane using the BN with adjacent lanes' detection only. In the second instance, we introduced the adjacent vehicles' detection in the BN. In all instances, we filtered the outcome of the BN with the HMM.

8.3.1/ QUANTITATIVE RESULTS

To evaluate our algorithm we compare the output of our LLL with the position of the ego-lane marking in the road. Accordingly, all the results obtained are summarized in Table 8.1.
The increment given by each module is clearly illustrated. Indeed, altogether cases the BN+ALD+AVD classification is more precise than the BN+ALD, implying that the addition of another information source will also increase the obtained accuracy. Furthermore, the HMM increases classification accuracy significantly in all cases as compared to the BN result. This is due to the model's inclusion of temporal cohesion. For example, the vehicle cannot shift from the leftmost to the rightmost lane in two successive frames. Another interesting point is the classification accuracy achieved for the 2-lanes datasets. We notice that the accuracy achieved with the BN+ALD+HMM is the same as the one achieved with the BN+ALD+AVD+HMM, which shows that the HMM was sufficiently robust in the first place and the addition of the adjacent vehicle's detector did not affect the results. Moreover, there are certain situations where vehicle detection is irrelevant, such as where the detected vehicle is on a nearby road. To solve this problem, we would have to determine the localization of the detected vehicle relative to the ego-vehicle, which is currently not the case since we use solely images as input.
Following further research into the incorrect results, it appears that the BN+ALD's false classifications was attributed to two major reasons:

- The lanes markings are not detected this can be explained if the lanes marking are missing or hidden by an object.

[1] The authors would like to acknowledge the authors of [109] for their help with the datasets

	$BN + ALD$	$BN + ALD + HMM$	$BN + ALD + AVD$	$BN + ALD + AVD + HMM$
$2 - lanes$	91.89%	99.00%	93.60%	99.00%
$4 - lanes$	67.36%	78.36%	74.67%	85.35%

Table 8.1: Classification accuracy for ego-lane determination. bn + ALD refers to the BN fed with the adjacent lanes' detection, bn+ALD+HMM indicates the HMM with the corresponding BN, bn+ALD+AVD refers to the BN feed with adjacent lanes and vehicles detection and bn+ALD+AVD+HMM refers to the HMM with the corresponding BN.

- The lanes markings are wrongly detected. For those cases, it shows the limitation of the used lane marking detector.

Finally, we manage to outperform the authors of work of [109] on the same datasets. Indeed, they achieved 77% correct classifications, where we were able to reach 85.35% on 9528 frames.

8.3.2/ QUALITATIVE RESULTS

In addition to the quantitative results presented, we present some qualitative results of the proposed LLL. In Figure 8.10 correct classification of the LLL is shown on different example images. As illustrated the output is coherent to the detection results. In the other hand, Figure 8.10 shows incorrect classification of the LLL. Taking into account these results:

1. In Figure 8.10e the lane marking of the leftmost lane has been not detected. In addition to that, no cars were detected in the left. Given these inputs, the LLL, chooses the second lane which is not incoherent.
2. In Figure 8.10f the lane marking of the exit lane have been detected misleading the output of LLL,
3. In Figure 8.10g and Figure 8.10h similar to the previous case, the lane marking for the exit lane has been detected leading to a false classification

In essence, even if the output of the proposed LLL is incoherent it is still coherent when taking into account the outputs. The work presented in this chapter have been published in IEEE Intelligent Vehicles Symposium (IV'19) [147].

8.4/ CONCLUSION

In this chapter, the problematic of LLL have been tackled, and a solution has been presented. The proposed solution is based on a probabilistic framework that is composed of two blocks. First, a modular BN is used to infer the ego-lane position from multiple inaccurate information sources. The flexibility of this BN is proven, by first, using only information from surrounding lane-marking detection, after, by adding adjacent vehicles detection information from a well-known DL network (YOLO [104]). Second, a designed

HMM is used to temporarily filter the outcome of the proposed BN. Indeed, this HMM translates the physical constraints of the vehicle when changing lanes. for example; the vehicle can not change lanes if these two lanes are not adjacent. Taking these considerations and exploiting the estimated ELL from Chapter 7, the HMM is able to enhance the proposed BN for lane determination.
Furthermore, real-world experimental results have been conducted to show the effectiveness of our algorithm. First, the algorithm has been tested on recorded images of national highway in the region of Clermont-Ferrand, where the roads have two lanes. Then, the performances were validated on more challenging scenarios and compared to an existing method, whose authors made their datasets public. Naturally, for each test we highlighted the modularity of the proposed framework by showing the increment provided by each block (BN+ lane detection, then BN+ vehicle detection +BN+HMM). In sum, our algorithm outperform the solution proposed by Ballardini *et al.* [109] since it provides more accurate classification (LLL 85.35% compared to 77%).
The next chapter will be dedicated to the validation of the end-to-end architecture on recorded datasets. The objective is to show the coherence of our proposed end-to-end solution in a real-world case.

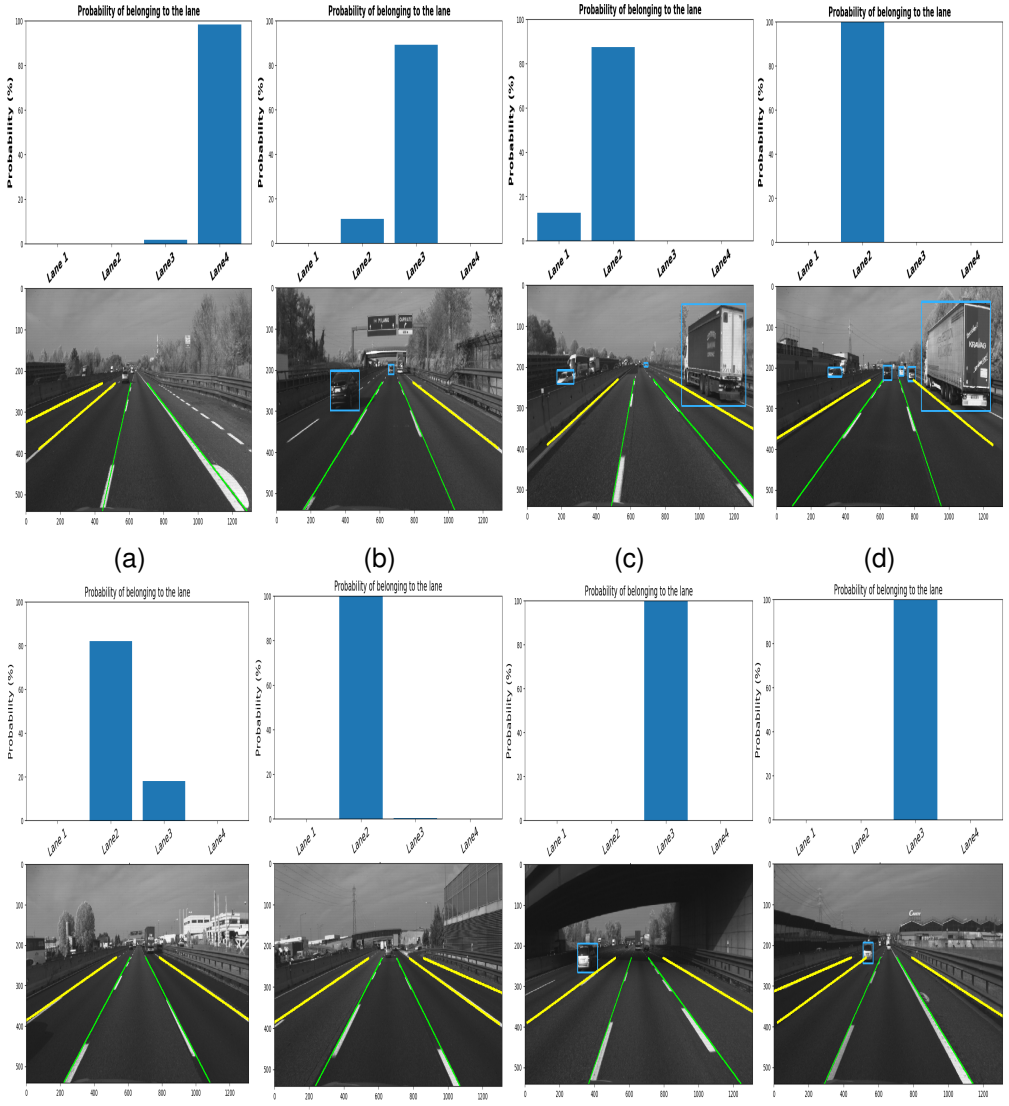


Figure 8.10: Examples of correct and incorrect ego-lane determination on the 4-lanes datasets.The first row only correct classification are given. On the other hand, the second row shows incorrect classification. In all cases, the ego-lane marking is shown in green and the adjacent vehicle are shown in Blue, finally, the yellow is used for the adjacent lane marking detected. An additional video is provided in shorturl.at/fjD06

.

9

END-TO-END COHERENCE RESULTS

.

Contents

9.1 Used acquisition platform 135

9.2 Overview of the dataset 135

9.2.1 Structure of the dataset 136

9.3 Performed experiments 136

9.4 Results and discussion 137

9.4.1 Functioning of the end-to-end algorithm 137

9.4.2 Results 140

9.5 Conclusion 144

The ego-localization problematic have been dissected into three parts: Road Level Localization (RLL), Ego-Lane Level Localization (ELL), and Lane-Level Localization (LLL). For each part, a solution has been proposed to address the localization requirements of each one. Furthermore, the proposed solutions are flexible and modular. Hence, each solution block can be used independently. Nevertheless, these solutions answer different localization's problematic. As a consequence, the way in which they are evaluated differs. Indeed, as stated in Chapter 5, there exist no datasets or metrics that can be used to evaluate at a stroke the whole end-to-end localization algorithm.

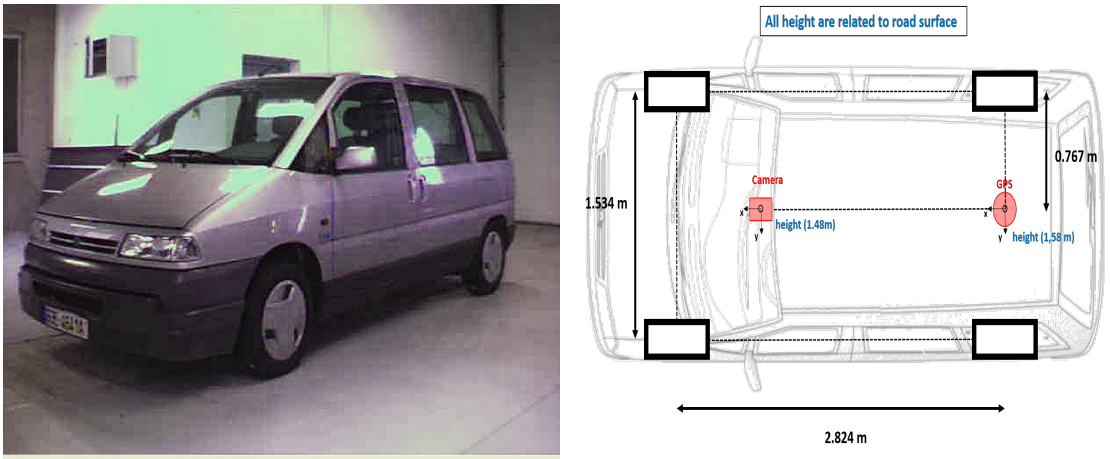
For these reasons, the strategy adopted in this work was to conduct from one side experiments validating the functioning of each proposed solution as well as their robustness to imperfect input data using real-world datasets. This first stage was done at the end of each chapter: in Section 6.4, Section 7.4, and Section 8.3.

On the other side, the overall end-to-end architecture is validated on two datasets that were collected in the region of Clermont-Ferrand. This chapter describes these datasets and the tools used to collect them. Finally, the end-to-end algorithm is evaluated using these datasets.

9.1/ USED ACQUISITION PLATFORM

Following its experience in the PROMETHEUS project (completed in December 1994) and its desire to develop an ambitious research program in the field of car driving, L'Institut Pascal (IP ex LASMEA) has been equipped since 1998 with an experimental vehicle named VELAC (Véhicule Expérimental du LASMEA pour l'Aide à la Conduite). The vehicle is a Citroën Evasion type. In addition to that, several sensors are mounted in VELAC to obtain information about the ego-vehicle and the environment (cf. Figure 9.1). The main sensors used for experiments are the following:

1. GPS receiver: a classical GPS receiver placed in the middle of the rear track
2. Gray Scale Camera: placed behind the rear-view mirror to capture the scene in front of the vehicle.
3. Inertial Measurement Unit (IMU): to capture the linear and angular velocities.
4. Wheel encoders: to estimate the distance traveled by the ego-vehicle.



(a) (b)

Figure 9.1: The vehicle used VELAC in Figure 9.1a, and the mounted sensor for the experiment in Figure 9.1b

9.2/ OVERVIEW OF THE DATASET

The experiment was conducted in the region of Clermont-Ferrand, more precisely on a highway containing 3 lanes. The choice of this road allows us to test every single block of our proposed algorithm. Furthermore, the road drove contains roads that have 3 lanes and contains more lane-changing scenarios comparing to other datasets (KITTI [70]). Therefore, the lane-level localization is more relevant and complex on this path since it contains more lanes and more lanes change.
The path traveled during this experiment is shown on the map as illustrated in Figure 9.2.

Figure 9.2: Path traveled in blue for a total of 5 km (image taken from Google)

9.2.1/ STRUCTURE OF THE DATASET

During this drive, the acquisition of all the data sensors provided by VELAC was done using the middleware ROS. The objective was to capture all the data recorded in the format that allows us to playback and visualize it. Hence, the dataset acquired is in the form of a bag. The bag is a file format in ROS which is used to store ROS message data. These message data are stored in different Topics that depend on the type of data. Furthermore, ROS offers also several tools for both recording and playback these data. As illustrated in Figure 9.3, a visualization of all topics that have been recorded in the bag file containing the dataset.

9.3/ PERFORMED EXPERIMENTS

The proposed end-to-end algorithm and each of the proposed modules (i.e, RLL, ELL, and LLL) have been implemented using Python. ROS provides a library Rospy that allows bridging between the Python code and the rosbag. This bridge allows us to test and evaluate the whole end-to-end algorithm on the real-world dataset and hence meet the specifications of this Ph.D.
The experiments performed concern all the three-components presented, namely, the RLL module, the ELL module, and the LLL module. In addition to that, we emphasize on the whole coherence between these modules and their interactions.

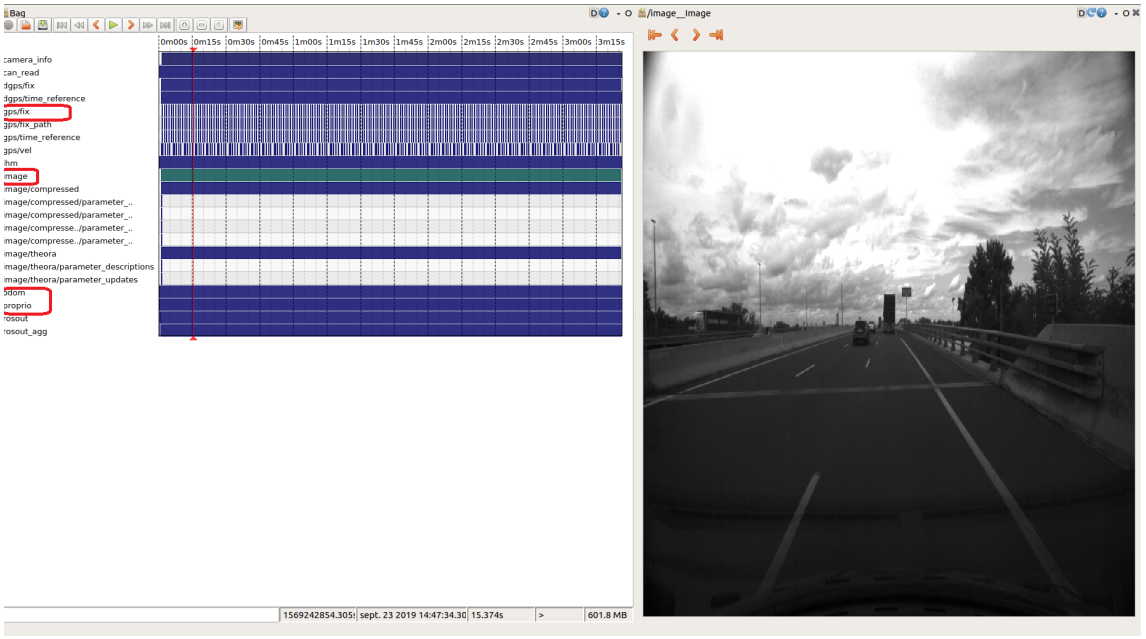


Figure 9.3: Topics stored in the bag using rqt_bag. Relevant topics are highlighted in red.

To summarize, the following evaluations have been performed:

- The road on which the vehicle is traveling (cf. the OSM Way), which has been manually annotated for each GPS frame,
- The lateral shift position of the ego-vehicle regarding its own lane. Although, no quantitative evaluation have been performed due to the lack of ground truth, qualitative evaluation have been performed, by tracking the lateral shift position over the time.
- The lane level localization evaluation, for which the position of the ego-lane have been manually annotated for each image frame.

In the following outcome of each block (RLL, ELL, and LLL) are presented.

9.4/ RESULTS AND DISCUSSION

Before presenting the outputs of our end-to-end algorithm, we start by presenting all the steps of the algorithm. These steps are supported by graphics that help to understand all the functioning of the algorithm.

9.4.1/ FUNCTIONING OF THE END-TO-END ALGORITHM

In the following algorithm's flow is presented on a canonical example:

1. first step is to locate the vehicle on the road using the low cost GPS. To do so, we start by retrieving the OSM's ways from the database with the adequate query using the Overpass API. The outcome is illustrated in Figure 9.4,

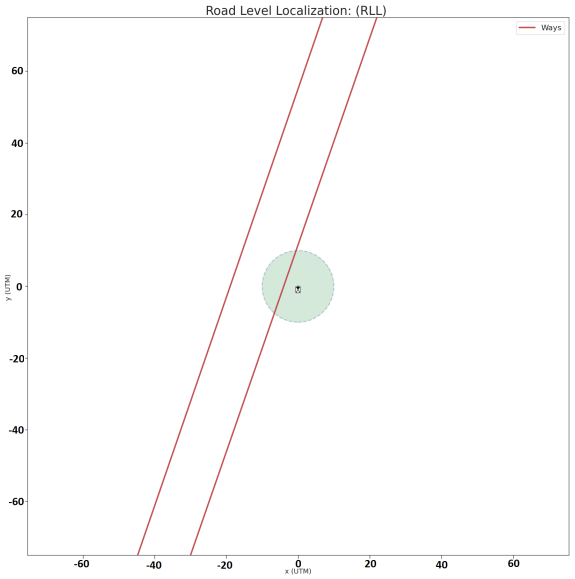


Figure 9.4: First step of the RLL with the display of the ways from OSM database. The drawing have been centered around the estimated vehicle position.

2. thereafter, the PMMA that has been presented in Chapter 6 is used in order to select the right way as presented in Figure 9.5. In addition to that, an estimation of the lanes number from OSM is performed,

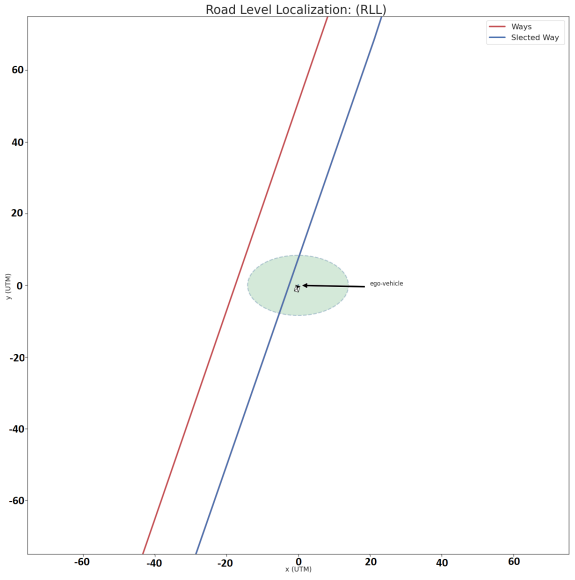


Figure 9.5: Selection of the Way on which the vehicle travels using the PMMA algorithm presented in Chapter 6.

3. once the road chosen, the road's parameter are estimated from the map. To do so, the road is projected in the vehicle space and fitted to the third degree polynomial. Hereafter, the ELL process starts by projecting the polynomial model into the sensor space (hence the camera image since no lidar is available in VELAC). The model projected take into account the uncertainties about the value of the parameters of the road, and is projected as illustrated in Figure 9.6,

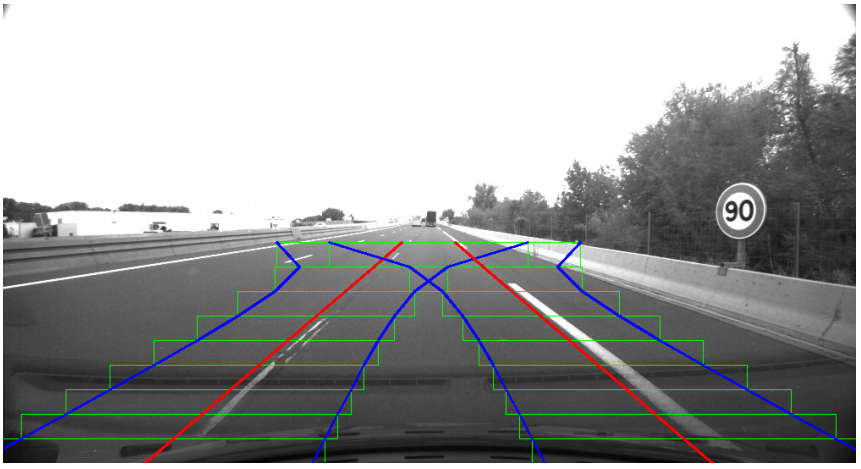


Figure 9.6: Projection of the polynomial model into the image space. The means values are represented in red, the interval of research in blue and the ROI in green.

4. subsequently, the RIDA algorithm is launched in order to detect the lane marking and hence estimate the road's parameter and the vehicle's state. The algorithm have been detailed in Section 7.3. Eventually, the ego-lane marking are detected as illustrated in Figure 9.7,

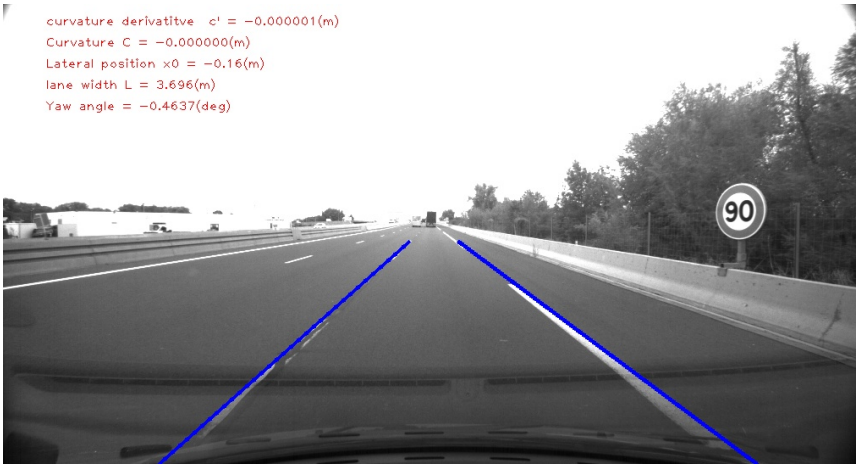


Figure 9.7: Detection of the ego-lane marking in blue, and estimation of the road's parameters.

5. once the ELL finished, the RLL algorithm is initiated. Beginning with adjacent lane detection, this detection benefits from the knowledge of the lanes width and the

number of lanes, in order to restrict the area of research in the sensor space as shown in Figure 9.8,

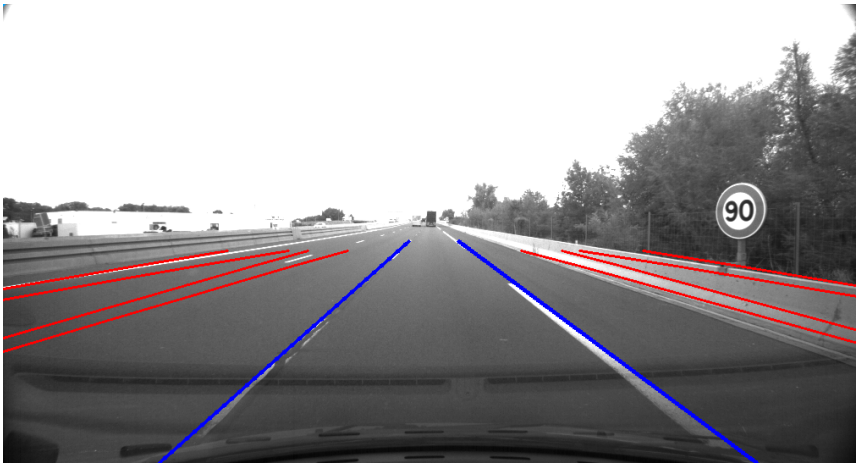


Figure 9.8: Area of research for the adjacent lane marking in red in which the detection will be performed.

6. meanwhile, a vehicle detector is launched in order to detect adjacent vehicle, in this work it relies on a YOLO network (cf. Section 8.2.2). The output of the both the adjacent lanes detection and vehicle detection is fed into a Bayesian Network that will infer the lane position in the road as shown in Figure 9.9.
7. for the next frame, a tracking step is performed on the estimated ego-lane marking and the probability to change lane is computed,
8. this probability is introduced in a HMM that will filter the output of the Bayesian network for the next LLL.

These steps are repeated for each frame of the collected dataset.

9.4.2/ RESULTS

The first result shown in this section corresponds to the output of the whole end-to-end algorithm (cf. Figure 9.10). The road in which the vehicle is traveling is shown on the upper left side of the figure. The ego-lane marking is shown with the red color and the lateral shift is kept tracked in the lower left side of the figure. Furthermore, adjacent lanes marking (correct or false) are shown in cyan, and the detected vehicles with the orange colors.

The second results shown corresponds to a sequence of outputs that highlights the functioning of the dynamic function of the algorithm (cf. Figure 9.11), which is one of the main contributions of this work. For clarity's sake, the images do not sequence in time. In other words, the time between the images is not the same. Indeed, the objective is to illustrate the behavior of the end-to-end algorithm on a sequence of the frame, and we want to empathize on key moments such as lane change.

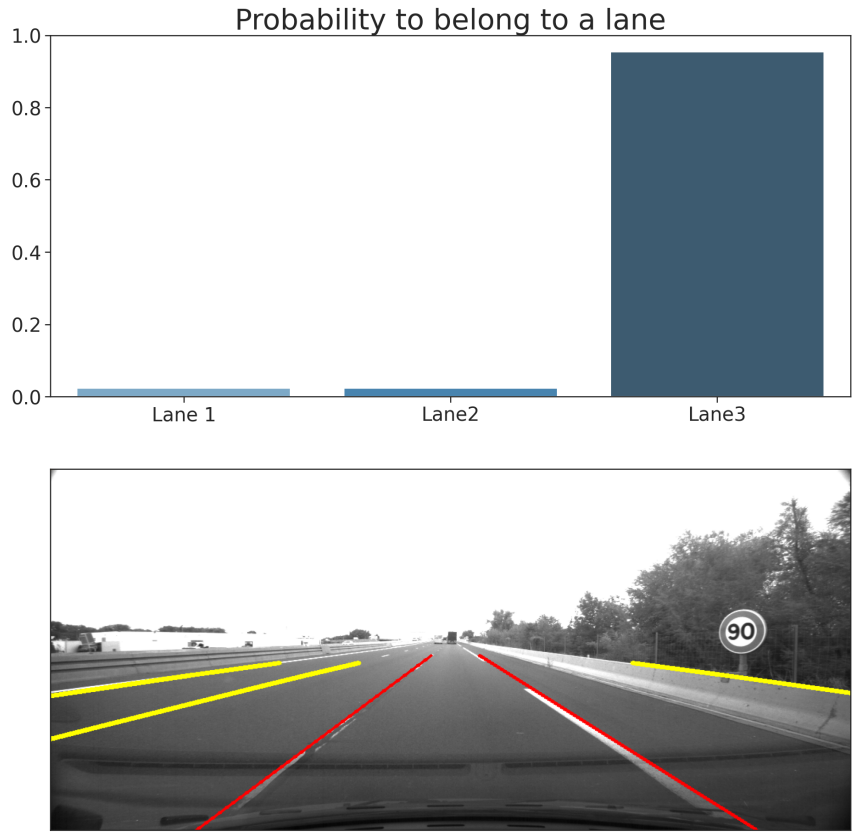


Figure 9.9: LLL performed taking into account the adjacent lane detection results in cyan, and the detector failure rate. The leftmost lane is noted $Lane_1$, while the rightmost lane is noted $Lane_3$

As illustrated in the mentioned figure, the ego-lane marking is kept tracked and the lane determination is correctly determined sequentially. For the first one, it is due to the tracking procedure that has been presented in Section 7.3.5. Concerning the change in lanes, it is governed by the modular probabilistic framework and especially the HMM designed in Section 8.2.3. In addition to that, the estimated lateral position allows us to determine lane change maneuvers that had occurred during the drive.
Nevertheless, in addition to these visual outputs, quantitative results have been noted. In Table 9.1 the reported results concern the RLL. Comparison between the addition of the HMM in the PMMA algorithms have been reported. The main goal is to point out the increment of this component in the proposed framework. As highlighted, the PMMA with the addition of the HMM improves the overall accuracy precision. This can be explained by the inclusion of the topology of the road network in the HMM. Indeed, the transition states are governed by the connectivity between roads.

Concerning the LLL, the results have been reported in Table 9.2. Furthermore, the increment provided by each detector (adjacent lane, vehicle detector) and the increment provided by the HMM is clearly highlighted in terms of accuracy. Indeed, altogether cases the addition of the HMM provides more accurate classification than the BN alone.

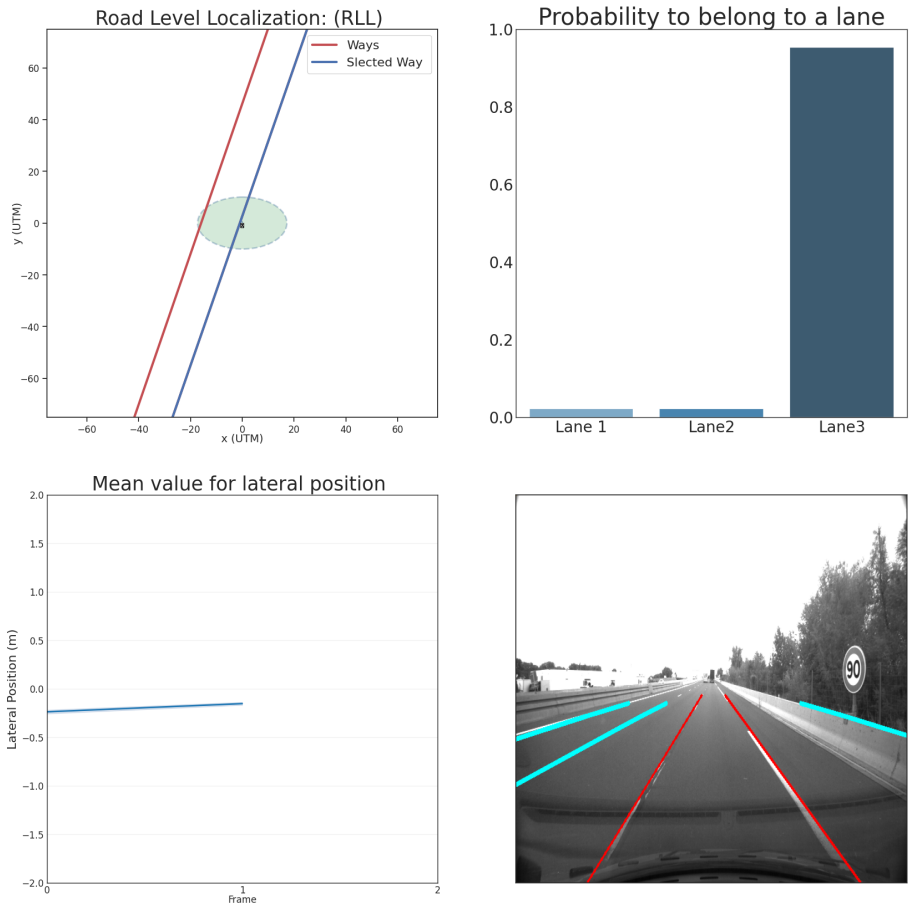


Figure 9.10: Example output of the proposed end-to-end algorithm. The RLL output is presented in the upper left of the image, the lateral shift of the vehicle in the lower left of the image, and the LLL in the right side of the image

	3-lanes
Number of frames	205
Correct MM Without HMM	99.51 %
Correct MM with HMM	100 %

Table 9.1: Map Matching results on the entire datasets without and with the HMM

	$BN + ALD$	$BN + ALD + HMM$	$BN + ALD + AVD$	$BN + ALD + AVD + HMM$
3-lanes	78.8%	86.8%	81.30%	90.90%

Table 9.2: Classification accuracy for LLL. BN+ALD refers to the BN feed with the adjacent lanes detection, BN+ALD+HMM indicates the HMM with the corresponding BN, BN+ALD+AVD refers to the BN feed with adjacent lanes and vehicles detection and BN+ALD+AVD+HMM refers to the HMM with the BN

Concerning the time execution, the presented results have been carried using Python 3.7 under a Dell G3 3579 Core i7 8th generation equipped with a NVIDIA GeForce GTX 1050 Ti. The computation time results presented in Figure 9.12 show that even if the entire algorithm was coded in Python, real-time implementation is possible. Indeed, the sum

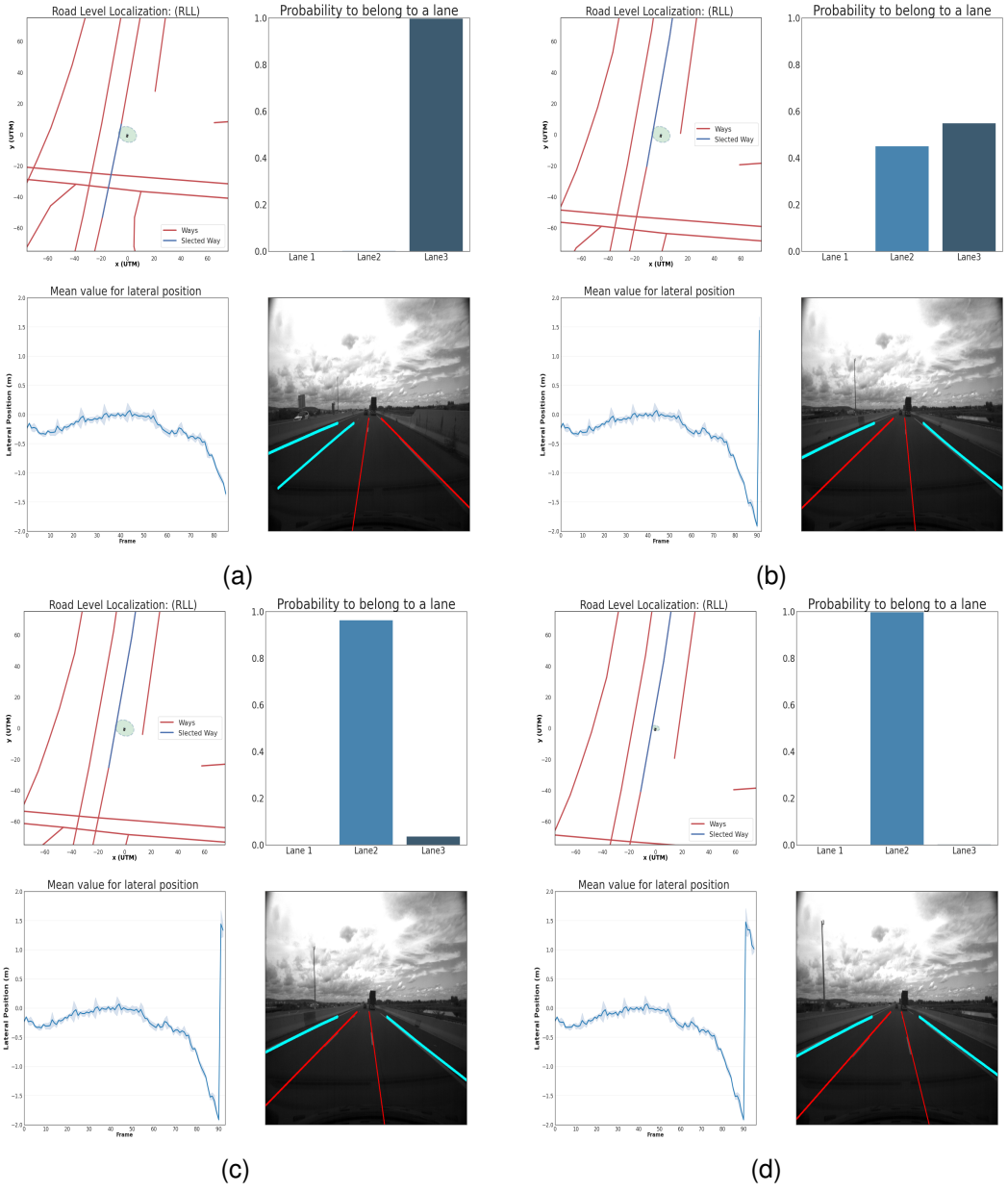


Figure 9.11: A sequence of outputs of the proposed end-to-end algorithm in a lane change scenario. In Figure 9.11a the vehicle is traveling in the third lane (the right-most lane). In Figure 9.11b the vehicle starts the left lane change, as a consequence the estimated lateral position go to $-L_w/2$. In Figure 9.11c and Figure 9.11d the vehicles completes the lane change and is now traveling the lane number 2.

of all parts of the ego-localization algorithm is under 500 ms (446.95 ms on average). In addition, if we glance in-depth at the time consumed in the RLL, we found that on average, it takes 238.16 ms to query the local server containing the OSM data. Furthermore, from the 143.67 ms dedicated to the LLL, 65.60 ms are spent on the YOLO detector. These results lead us to assume that implementation on C/C++ will divide the calculation time by 10 and hence will be relevant for real-time application. It is to be noted that the results of the experiment stated above are available through this link : shorturl.at/mGKT4.

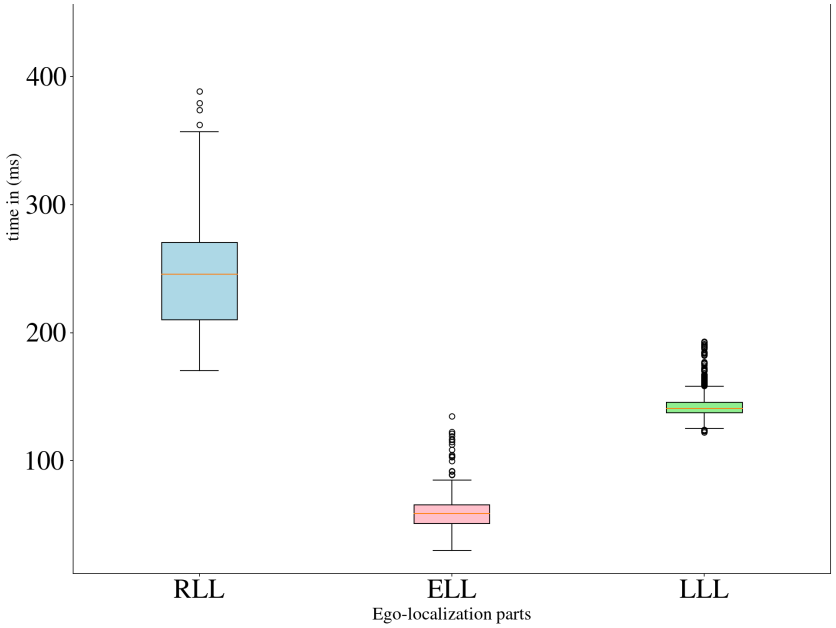


Figure 9.12: Computation time in ms for each part of the presented algorithm: RLL in Light-blue, ELL in pink and LLL in light-green, with mean values of $244.53ms$ for the RLL, $58.74ms$ for the ELL and $143.68ms$ for the LLL. Since the computation time for the ELL and LLL depends on the observability conditions of the markings lanes in the image, the dispersion in time is much bigger than the RLL.

9.5/ CONCLUSION

In this chapter, the end-to-end localization algorithm has been validated and evaluated on a collected data. This collected dataset is made available for other research through this link: shorturl.at/lAR28. In addition to that, for the different experimentation conducted, videos are available through this link: https://shorturl.at/gHRT7.
The evaluation of the proposed end-to-end localization algorithm conducted on the dataset attested to its effectiveness. Furthermore, this chapter aimed to emphasize on the global coherence of the proposed solution from RLL to the LLL modules. Indeed, these modules have been separately evaluated and tested (cf. Section 6.4, Section 7.4, and Section 8.3). Nevertheless, the global coherence of the proposed architecture has well illustrated in Section 9.3.
Moreover, even if experiments shown in this work have validated several aspects of the proposed end-to-end localization algorithm, it remains several aspects that could be improved. Indeed, the time computation has to full-fit real-time application. To do so, a C++ implementation has to be done. Besides, the presented dataset do not contain a lidar sensor. Hence, it will be interesting to test the whole architecture on a vehicle equipped with this type of sensors. Finally, it will be interesting to test the presented architecture on more challenging roads (i.e., highways in North American country with 6 lanes and more). Further perspectives are cited in the following chapter.

GENERAL CONCLUSION AND PERSPECTIVES

GENERAL CONCLUSION

This Ph.D. thesis work addresses several aspects related to the localization of an autonomous vehicle, mostly in highway scenarios. This localization has been partitioned into three components, namely the Road Level Localization (RLL) which is the knowledge of the road on which the vehicle is traveling on, the Ego-Lane Level Localization (ELL) which is related to the lateral position of the ego-vehicle in his own lane, and Lane-Level Localization (LLL) which is the knowledge of the lane's index on which the vehicle is traveling on. For each component, an appropriate solution have been proposed that satisfies the localization's requirement of each component. Furthermore, these solutions are the cornerstone of a unified sequential framework entitled "end-to-end probabilistic framework for ego-localization" which is the main contribution of this Ph.D. thesis. In the following, a recapitulation of the main subjects discussed in each of the chapters composing this manuscript.

A global introduction on the past and present of the autonomous vehicle in addition to the current and future challenges that are still unsolved for autonomous vehicles have been given. With that background, context and motivation of our research work have been discussed in Chapter 1. The main objective is to propose a unified localization framework for ego-vehicle localization that covers all the localization requirements from RLL to LLL. Under these considerations, the first part of the manuscript was dedicated to state-of-the-art review related to the autonomous vehicle localization. Subsequently, this review has been divided into four parts, each part was studied in a chapter.

Starting with Chapter 2, on which literature about Road Level Localization (RLL) systems have been reviewed. Accordingly, the majority of the RLL techniques relies on Map-Matching (MM) methods. These methods are clustered to whether they are deterministic or probabilistic. Deterministic methods are straightforward techniques that do not require complex computation. As a result, the running time of the algorithms falling into this category is very low compared to other methods. However, these methods are very tributary to the quality of the raw data used. On the other hand, probabilistic have power in their capability to handle uncertainties and matching break situations. The backlash is that they are complex and require more computations. These conclusions suggested that a hybrid architecture will benefit from the strength of the two approaches.

Afterward, Chapter 3 was dedicated to ELL literature review. A focus was given to ELL techniques that are based on the detection of the ego-lane marking. Accordingly, these techniques are grouped into two categories. The first category is the model-based approaches. This has a strong ability to detect the ego-lane marking in various scenarios. The sequential pipeline architecture of the methods that fall under this category allows a better partition of the ego-lane marking detection, with each part being responsible for a

specific task. Furthermore, it allows better system failure identification and enables them to improve or incorporate new functionalities that were not supported in the initial design without requiring significant modifications. However, the generalization of such methods is complicated and challenging for highly complex road scenarios. On the other side, the second category called the monolithic learning approach is based on Neural Network perform better when a model can not be formalized or is not available. Nevertheless, these methods require a learning phase preliminary to the deployment of the network. Therefore, their capabilities are highly affected to the quality of the data which is limited in quantity and is usually gathered for a specific task and for a specific configuration (e.g, same city, same camera, same weather condition). Taking under account these considerations, we came out that model-driven approaches are more relevant to our range of application, which is the highway roads scenarios.
Thereafter, Chapter 4 reviewed techniques related to LLL. Accordingly, two paradigms are used. The first one is the knowledge of the lateral position of the autonomous vehicle with respect to the road. In this paradigm, lane-level map-matching algorithms are used to match the estimated position of an ego-vehicle with a map. Generally speaking, the type of map used for this kind of task is the Mesoscale map. Contrary to the map-matching methods presented in Chapter 2, the lane-level map-matching algorithm is more robust since they deal with more complex and ambiguous scenarios. Moreover, this paradigm suffers from its dependency on the availability of this kind of map. Indeed, these maps are cost-intensive and not open-source. The second paradigm articulates the LLL as a classification problem. Therefore, it relies solely on the detection of visual landmark, these detections are then fed into high-level fusion frameworks that will infer the lane localization of the ego-vehicle. In that sens, this paradigm is independent of the utilization of sophisticated maps, and hence is more compatible with a solution that relies purely on an open-source map. The last chapter (cf. Chapter 5) of this part was dedicated to the existing ways to evaluate such algorithms. It appears that most researchers use collected datasets to evaluate their algorithms. By doing so researchers can evaluate their algorithms on real-world scenarios without taking the risk of deploying their solutions on real cars. In addition to that, researchers can evaluate and compare with each other their techniques, which creates a constructive and healthy competition among them. Therefore, a literature review of the different available datasets that can serve to evaluate the end-to-end localization algorithm have been presented. In addition to that, metrics that are used in the majority of benchmarks have been discussed. This study leads us to the conclusion that there exist no real dataset that can be used in order to evaluate the full pipeline of our proposed solution.
In the second part of the manuscript, the proposed approach noted as "end-to-end probabilistic framework for ego-localization" have been detailed and evaluated.
The first chapter of this part (cf. Chapter 6) introduced an architecture that combines two constructed modules to determine the RLL. The first module is in charge of removing any roads that are incompatible with the vehicle configuration in terms of distance and orientation criteria. Furthermore, a speed limit criterion was taken into consideration. Thereupon, if the number of roads candidates is greater than one, a selection strategy is designed. Accordingly, three probabilistic criteria were investigated in order to model this selection strategy. The mathematical formalization has been given for each of these. Furthermore, a second supplementary block has been added based on the output of the first block in order to clear up ambiguous situations and account for history navigation. In that sens, a HMM was designed to model the road topology and the vehicle's cinematic. Details about its modeling has been given. In the second chapter of this

part (cf. Chapter 7), an ELL solution has been detailed based on the detection of ego-lane marking for both lidar and camera images. The proposed framework is based on information-driven approach in which the objective guide the recognition process. Therefore, a Recursive Information-Driven Algorithm (RIDA) has been described in detail. To minimize the computation's complexity, this algorithm uses entropy features to select the most relevant data in the sensor space. Furthermore, the mathematical formalization of each RIDA component has been explained. Afterward, in Chapter 8 a probabilistic framework that solves the LLL problematic was presented. It is composed of two blocks. First, multiple inaccurate information sources are used to infer the ego-lane location using a modular semi-fixed BN. The flexibility of this BN is proven, by first, using only information from nearby lane-marking detection and then incorporating adjacent vehicle detection information. Second, the BN's output is temporarily filtered using a designed HMM. This HMM does also translate the vehicle's physical restrictions when shifting lanes. Indeed, the vehicle would not be able to change lanes if the two lanes are not adjacent. Taking this consideration and exploiting the estimated ELL from Chapter 7, this problematic framework is able to correctly identify the lane on which the vehicle is traveling. Moreover, real-world experimental results on different datasets justifying the effectiveness of each block RLL, ELL, and LLL have been given in the end of each chapter. Finally, the last chapter (cf. Chapter 9) was intended to illustrate the behavior of the whole end-to-end localization algorithm. In addition to that, a collected dataset, that is made available for other research through this link: shorturl.at/lAR28, has been presented. Further to that, an evaluation of the proposed end-to-end localization algorithm on this dataset was given. It attested on the global coherence of the interconnected module composing the whole algorithm. In addition to that, metrics have been used to illustrate the effectiveness of the latter.

PERSPECTIVES

The research work outlined in this dissertation has shown promising results considering the overall end-to-end localization architecture presented and problematics solved by the latter. Motivated by these promising results, we believe that it remains several aspects that could be improved and other to be investigated. The following is a list of the most relevant future works.

Improve the overall architecture Notwithstanding the advantages of the proposed end-to-end ego-localization algorithm, it remains several aspects that can be improved:

- There exists no recovery strategy or feedback between modules, which means that the RLL modules influences the ELL which influence the LLL. However, the latest modules do not question the higher modules. For future work, a monitoring module should be added to enhance the proposed framework.
- The RLL module (cf. Chapter 6) is highly dependent on the estimated vehicle's heading. Indeed, the nature of our PMMA algorithm depends on a good estimation of the vehicle heading for choosing the right Way. In addition to that, the uncertainty of the GPS receiver are modeled with a Gaussian distribution. This modeling does not always cover the uncertainties of the GPS that may come with a bias error due to

non-white noises. Therefore, a better modeling of such error should be considered in future works.

- In the ELL part (cf. Chapter 7), the polynomial representation of the road is very efficient for highway roads. However, this representation does not hold for large value of curvature. Hence, this model will not suit perfectly the ego-lane marking to sharp curves (roads that has 'S' shape). Therefore, non-parametric model should be taken into account for these kinds of situations.
- In the LLL part (cf. Chapter 8), the vehicle detector is used in 2d image space. Hence, no 3D representation of the adjacent vehicle are utilized. It will be interesting to use a 3D representation (i.e, Lidar data)
- Finally, for future work it would be interesting to add detectors from different sensors, i.e., lidar, radar to enhance the proposed framework. Indeed, the modular nature of the proposed algorithm allows such addition.

Enhanced the existing with DL techniques DL are de facto the main solutions used in computer science and especially in AVs. Indeed, these methods that use these techniques have been widely adopted for task like object recognition (i.e, lane marking, vehicle). Although, we did use a vehicle detector for the LLL, we believe that there is room for more DL to be incorporated in a localization system for AVs. The idea is to fuse the power of the DL for low-level tasks like lane detection, vehicle detection, with the ability of the model-driven approach to model the physical constraints of the real-world. The fuse of the two worlds will allow a better detection of traffic road object, while keeping the global coherence of the road scene. The concept of this new architecture is illustrated in Figure 9.13

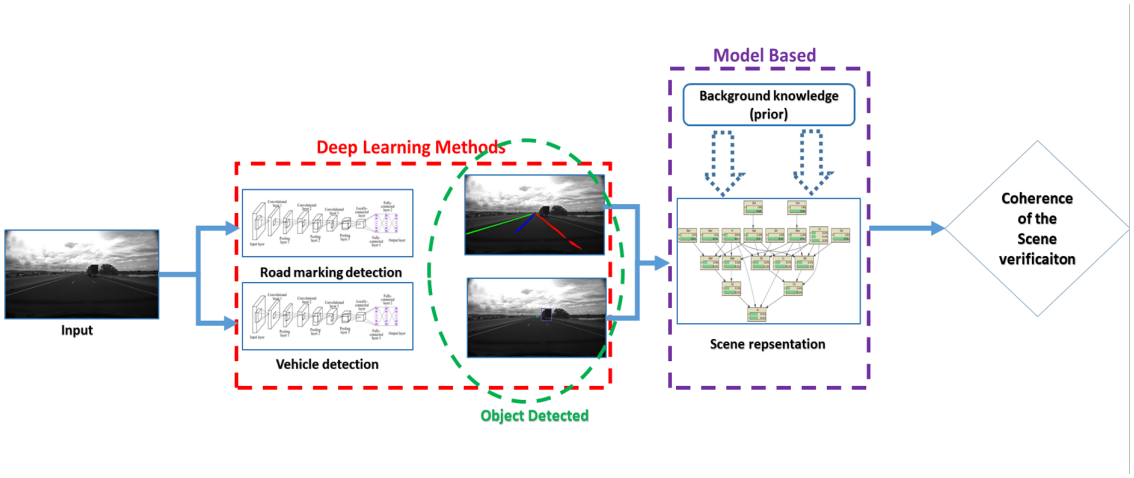


Figure 9.13: Concept of the addition of DL with model-based technique in order to have a better road scene representation and hence a better ego-localization.

Real time experiments Although several experiments have been conducted and presented in this manuscript (cf. Chapter 9), real time experiments may be the Achilles' heel

of the validation phase of the proposed framework. However, the Institut Pascal recently acquired two Renault Zoé vehicles (as illustrated in Figure 9.14). One of them, will serve as an acquisition vehicle in order to collect dataset, while the other one will be robotized in order to test control/command framework. In that sens, it will be interesting to test our algorithm on these vehicles. However, before performing such a task, the portability of the algorithm to C++ have to be done. This task has already been started by an intern in 2020. Nevertheless, the algorithm's performance could be evaluated in more challenging weather conditions (i.e, snow, rain).

Figure 9.14: The new experimental vehicle acquired by Insitut Pascal

Extend the range of application of the overall localization architecture In this thesis work, we emphasize on highway and national roads have been treated. Thereupon, it would be worthwhile to investigate other scene scenarios in order to validate the portability of the proposed localization system. Therefore, the following is a list of some relevant road scene scenarios:

- Urban scenarios: it would be interesting to test the algorithm in urban scenarios. Indeed, in such cases the roads are crowded with cars, which make the detection of lane marking hard. In addition to that, the RLL part is also complicated by the GNSS errors due to the multiple path.
- Rural roads: In which there is no lane marking. For these scenarios, the difficulty appears when detecting lane marking. Indeed, for some rural roads, there is no paint on the asphalt.
- Roundabout, intersections, and toll stations.

10

ANNEX

10.1/ HIDDEN MARKOV MODEL(HMM)

This section introduces the HMM, with application related to our work, those classifications are based on work of Russel *et al.* [66].
Hidden Markov model or HMM is a temporal probabilistic model in which the state of the process is described by a single discrete random variable. The possible values of the variable are the possible states of the world.
The HMM belongs to the family of Dynamic Bayesian Network (DBN). In order to construct a HMM (or a DBN) we have to specify three components:

- the prior distribution over the state variables $P(X_0)$
- the transition model, which specifies the probability distribution over the latest state variables, given the previous values, that is, $P(X_t|X_{0:t1})$. However, a problem arises when the set $X_{0:t1}$ is unbounded. To solve this problem, the **Markov Assumption** is made. This assumption states that the current state depends only on a finite number of previous states. In the simplest case is the first-order Markov process, in which the state depends only on the precedent state, meaning that $P(X_t|X_{0:t1}) = P(X_t|X_{t1})$. This conditional probability is called **Transition probability**
- the sensor model $P(E_t|X_t)$ or the observation model. Even though the states of a HMM not visible, they are observable. Therefore, there is a relation between the hidden elements of the state and the observations, this relation is referred to as an emission probability $P(E_t|X_t)$

With that, we have a specification of the HMM we can perform several tasks like [66]: filtering, prediction, smoothing, most likely explanation, and learning. In our work, we are mostly interested with filtering process.
Rather than going back thought the whole history of observations after each update, a useful filtering algorithm must keep a current state estimation and update it. In other words, given the filtering results at time t, the filter algorithm has to compute the results for $t+1$ using new observations (evidences) e_{t+1}. This task is called **recursive estimation** where the objective is to find some function f that satisfies the following formula:

$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = f\left(\mathbf{e}_{t+1}, \mathbf{P}(\mathbf{X}_t \mid \mathbf{e}_{1:t})\right) \tag{10.1}$$

By arranging this formula we can have :

$$\begin{aligned}&\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}, \mathbf{e}_{t+1}) \quad \text{(dividing up the evidence)}\\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}, \mathbf{e}_{1:t}) \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}) \quad \text{(using Bayes'rule)}\\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}) \quad \text{(by the sensor Markov assumption).}\end{aligned} \tag{10.2}$$

With α is a normalizing constant used to make probabilities sum up to 1. The term $P(X_{t+1}|e_{1:t})$ represents a one-step prediction of the next state, and the first term updates this with the new evidence. The term $P(e_{t+1}|X_{t+1})$ is directly obtainable sensor model (emission model). Now we obtain the one-step prediction for the next state by conditioning on the current state X_t:

$$\begin{aligned}&\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \textstyle\sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t, \mathbf{e}_{1:t}) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})\\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \textstyle\sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t}) \quad \text{(Markov assumption).}\end{aligned} \tag{10.3}$$

This formulation gives us the desired recursive formulation. In order to show the functioning of this formulation an example is given.
Imagine a security guard assigned to a top-secret underground facility. He want to know if it's rainy outside, but his only access to the outer world is when he sees the director come in with or without an umbrella each morning. For each day t, the set E_t thus contains a single evidence variable U_t (whether the umbrella appears), and the set X_t contains a single state variable R_t (whether it is raining). A graphical illustration is given in Figure 10.1.

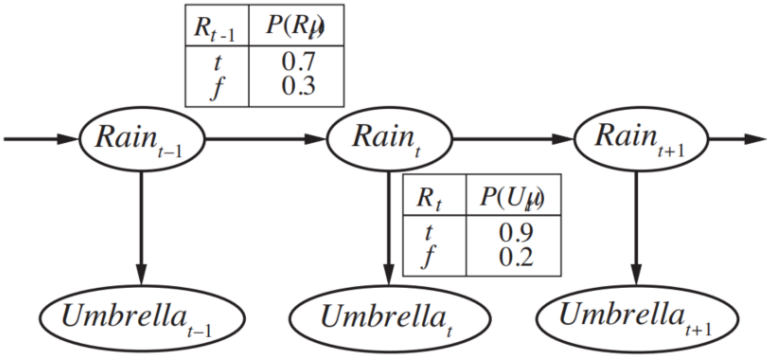


Figure 10.1: Example of the umbrella example, the transition model is $P(Rain_t|Rain_{t1})$ and the sensor model is $P(Umbrealla_t|Rain_t)$ (image credit [66]

To illustrate the filtering process we compute $P(R_2|u_{1:2})$ as follows:

- On day 0, there are no observations, therefore an equiprobability of chance of raining, that is, $\mathbf{P}(R_2 \mid u_{1:2})$
- day 1, the director appears with an umbrella, so $U_1 = true$. Therefore $\mathbf{P}(R_1)$ is computed as follows:

$$\mathbf{P}(R_1) = \sum_{r_0} \mathbf{P}(R_1 \mid r_0) P(r_0) \tag{10.4}$$

$$= \langle 0.7, 0.3 \rangle \times 0.5 + \langle 0.3, 0.7 \rangle \times 0.5 = \langle 0.5, 0.5 \rangle \tag{10.5}$$

Then the update step simply multiplies by the probability of the evidence for $t = 1$ and normalizes:

$$\mathbf{P}(R_1 \mid u_1) = \alpha \mathbf{P}(u_1 \mid R_1)\mathbf{P}(R_1) = \alpha\langle 0.9, 0.2\rangle\langle 0.5, 0.5\rangle \tag{10.6}$$
$$= \alpha\langle 0.45, 0.1\rangle \approx \langle 0.818, 0.182\rangle \tag{10.7}$$

- day 2,the director appears with an umbrella, so $U_2 = true$. Following the same strategy:

$$\mathbf{P}(R_2 \mid u_1) = \sum_{r_1} \mathbf{P}(R_2 \mid r_1) P(r_1 \mid u_1) \tag{10.8}$$
$$= \langle 0.7, 0.3\rangle \times 0.818 + \langle 0.3, 0.7\rangle \times 0.182 \approx \langle 0.627, 0.373\rangle \tag{10.9}$$

updating it with the evidences from $t = 2$:

$$\mathbf{P}(R_2 \mid u_1, u_2) = \alpha \mathbf{P}(u_2 \mid R_2)\mathbf{P}(R_2 \mid u_1) = \alpha\langle 0.9, 0.2\rangle\langle 0.627, 0.373\rangle \tag{10.10}$$
$$= \alpha\langle 0.565, 0.075\rangle \approx \langle 0.883, 0.117\rangle \tag{10.11}$$

10.2/ BAYESIAN NETWORK (BN)

In this appendix a summary of Bayesian Network is given. Before, a quick introduction to the probability theory must be given. In all cases, this annex is based on the work of Russel *et al.* [66].

10.2.1/ PROBABILITY PROPERTIES

According to Russel *et al.* [66], a probability is a measure over a set of events that satisfies three axioms:

- The measure of each event is between 0 and 1. That being said, it is written as follows $0 \leq P(X = x_i) \leq 1$, with X being a random variable representing the event and x_i are the possible value of this random variable X. In most works, random variable are denoted by uppercase and their values by lowercase letters.
- The measure of the whole set is 1 i.e., $\sum_{i=1}^{n} P(X = x_i) = 1$.
- The probability of a union of independent events is the sum of the probabilities of these individual events. We can write $P(X = x_1 \vee X = x_2) = P(X = x_1) + P(X = x_2)$.

Generally, $\mathbf{P}(X)$ is used to denote the vector of values $\langle P(X = x_1), \ldots, P(X = x_n)\rangle$. In addition, we use $P(x_i)$ to denote $P(X = x_i)$ and $\sum_x P(x)$. for $\sum_{i=1}^{n} P(X = x_i)$.
The joint probability for two random variables A and B denoted $P(A, B)$ is defined as follows:

$$\mathrm{P(A, B) = P(A)P(B)} \tag{10.12}$$

On the other hand, their conditional probabilityP(B|B), also defined as the posterior probability (or just “posterior” for short) is mathematically defined as follows:

$$\mathrm{P(A|B)} = \frac{\mathrm{P(A, B)}}{\mathrm{P(B)}} \tag{10.13}$$

This definition of conditional probability can be written in a different form called the **product rule**:

$$P(A, B) = P(A|B)P(B) \tag{10.14}$$

Taking into account these definitions, two random variables A and B are said to be independent, ifif:

$$\mathbf{P}(A|B) = \mathbf{P}(A) \quad \text{or} \quad \mathbf{P}(B|A) = \mathbf{P}(B) \quad \text{or} \quad \mathbf{P}(A, B) = \mathbf{P}(A)\mathbf{P}(B) \tag{10.15}$$

10.2.1.1/ BAYES' RULE

The more general case of Bayes' rule for multi-valued variables can be written as follows:

$$\mathbf{P}(B|A) = \frac{\mathbf{P}(A|B)\mathbf{P}(B)}{\mathbf{P}(A)} \tag{10.16}$$

Many modern AI schemes for probabilistic inference are based on this simple equation. On the surface, this Bayes' rule does not seem very useful. Indeed, it computes the single term $\mathbf{P}(B|A)$ in terms of three parts: $\mathbf{P}(A|B)$, $\mathbf{P}(B)$ and $\mathbf{P}(A)$, Which seems to be a step backwards. However, in practice there are many cases where we do have good probability estimates for these three numbers and need to compute the fourth. Often, we perceive as evidence the *effect* of some unknown *cause* and we would like to determine that cause. In that case, Bayes' rule becomes:

$$\mathbf{P}(\ \mathit{cause}\ |\ \mathit{effect}\ ) = \frac{\mathbf{P}(\ \mathit{effect}\ |\ \mathit{cause}\ )\mathbf{P}(\ \mathit{cause}\ )}{\mathbf{P}(\ \mathit{effect}\ )} \tag{10.17}$$

The conditional probability $\mathbf{P}(\ \mathit{effect}\ |\ \mathit{cause}\ )$ quantifies the relationship in the causal direction, whereas $\mathbf{P}(\ \mathit{cause}\ |\ \mathit{effect}\ )$ describes the diagnostic direction.
In medical diagnosis, for example, conditional probability on causal correlations are often used, implying that the doctor is aware of the relationship $\mathbf{P}(\ \mathit{symptoms}\ |\ \mathit{disease}\ )$ and want to derive a diagnosis, $\mathbf{P}(\mathit{disease}\ |\ \mathit{symptoms}\ )$.
Furthermore, Bayes' rule is also useful dealing with combining independent evidence. Imagine having one *cause* for several *effect*. As a result, the full joint distribution can be written as follows:

$$\mathbf{P}\left(\ \mathit{Cause}\ ,\ \mathit{Effect}_1, \ldots,\ \mathit{Effect}_n\right) = \mathbf{P}(\ \mathit{Cause}) \prod_i \mathbf{P}\left(\mathit{Efect}_i|\ \mathit{Cause}\right) \tag{10.18}$$

10.2.2/ BAYESIAN NETWORKS

A Bayesian network is a Directed Acyclic Graph (DAG) in which the quantitative probability information is annotated to each node. The full specification is:

- Each node is a random variable, which could be discrete or continuous.
- A set of links or arrows are directed to connect pairs of nodes. If the arrow is from the node X to node Y, it is said that X is Y's parent. The intuitive meaning of an arrow is typically that X has a direct influence on Y.
- Each node X_i has a conditional probability distribution $P(X_i|Parents(X_i))$ that quantifies the effect of the parents on the node

Viewed as a piece of “syntax”, a Bayesian network is a DAG with some numeric parameters attached to each node. For example, you have recently mounted a new burglar alarm system in your house. It is reasonably effective at detecting a break-in, but it also reacts to small earthquakes on occasion. John and Mary, two of your neighbors, have offered to contact you at work if they hear the alarm. When John sees the alarm, he most often calls, but he sometimes confuses the phone ringing with the alarm.Mary, on the other hand, enjoys noisy music and often ignores the alarm. We’d like to quantify the likelihood of a robbery based on the facts of who has or has not called. This example is illustrated on Figure 10.2.

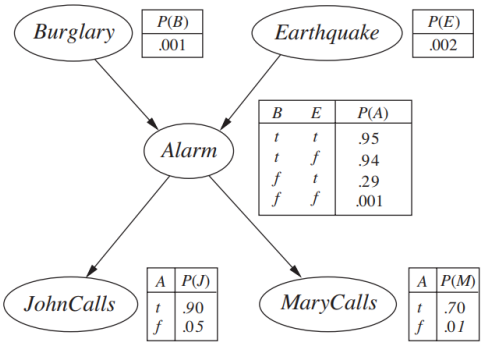


Figure 10.2: Example of a Bayesian network, showing both the topology and the conditional probability tables (CPTs). In the CPTs, the letters B, E, A, J, and M stand for Burglary, Earthquake, Alarm, JohnCalls, and MaryCalls, respectively (image credit [66]

The conditional probabilities of a node X_i noted as: $\mathbf{P}(X_i|\text{Parents}(X_i))$ are summarized in a Conditional Probability Table (CPT) like the one illustrated in Figure 10.2. One way of defining what the network is about is to describe how it reflects a specific joint distribution across the entire Variables defined as fellows::

$$\mathbf{P}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathbf{P}(x_i|\text{parents}(X_i)) \tag{10.19}$$

This equation governs what a Bayesian Network means. Using this equation and the previous equations allows us to calculate all the probability of each node presented in the Network.

10.3/ PRODUCT OF THE GAUSSIAN

The objective is as follows: Let’s assume that we have an area of presence of a given robot or vehicle, this in the form of a pose $\underline{X}$ centered on $\overline{X}$ and its covariance $\mathbf{C_x}$, such that $\underline{X} \sim \mathcal{N}(\overline{X}, \mathbf{C_x})$. In addition to that, let’s assume we have a known segment $[AB]$.The objective is to compute the probability that the segment belongs to this Gaussian zone (cf. Figure 10.3).

To do so, we represent the segment $[AB]$ by an ellipse itself defined by a covariance matrix $\mathbf{C_s}$.
By definition, a covariance matrix $\mathbf{C}$ has the following properties:

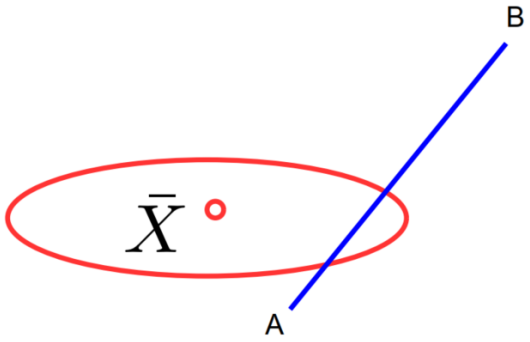


Figure 10.3: We want to know the probability that the segment [AB] belongs to the Gaussian zone

- the eigenvectors of this matrix are the axes of the ellipse,
- for a one standard deviation ellipse, the square roots of the eigenvalues of the matrix are the lengths of the axes of this ellipse

The objective is to find the covariance matrix so that its principal eigenvector is the considered segment.
Given a matrix $\mathbf{A}$ with eigenvalues λ_i and eigenvectors V_i with $i \in [1, N]$ (N is the dimension of the matrix). Because of the properties of the eigenvectors and eigenvalues, we have for a pair (λ_i, V_i):

$$AV_i = \lambda_i V_i$$

If we put all these vectors in a matrix $\mathbf{V}$ and the eigenvalues in a matrix $\mathbf{L}$ such that

$$V = V_1, ..., V_N \quad and \quad L = \begin{pmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & 0 & ... \\ ... & ... & ... & ... \\ 0 & ... & ... & \lambda_N \end{pmatrix}$$

Thus:

$$\mathbf{AV} = \mathbf{VL} \tag{10.20}$$

Therefore, we can find the matrix A knowing its eigenvectors and eigenvalues:

$$\mathbf{A} = \mathbf{VLV^{-1}}$$

To determine of a covariance matrix from a segment $[AB]$ with points $A = (x_A, y_A)^T$ and $B = (x_B, y_B)^T$. The vector $\underline{AB}$ will be given by :

$$\underline{AB} = \begin{pmatrix} x_b - x_a \\ y_b - y_a \end{pmatrix} = \begin{pmatrix} x_{AB} \\ y_{AB} \end{pmatrix}$$

Therefore the covariance matrix $\mathbf{C_s}$ has to satisfies the following:

- $\mathbf{C_s}$ has principal eigenvector (largest eigenvalue) $\underline{AB}$. Thus:

$$V_1 = \underline{AB}$$

- the eigenvalue corresponding to this vector $\underline{AB}$ will be the square of half the length AB of the segment $[AB]$. Thus
$$\lambda_1 = \frac{AB^2}{4}$$
- the second eigenvector V_2 must be orthogonal to V_1, so we will choose :
$$V_2 = \begin{pmatrix} -y_{AB} \\ x_{AB} \end{pmatrix}$$
- the eigenvalue λ_2 corresponding to V_2 will correspond to half the thickness th_s of the segment, we will give it a small value. Hence:
$$\mathbf{V} = \begin{pmatrix} x_{AB} & -y_{AB} \\ y_{AB} & x_{AB} \end{pmatrix} \quad \text{and} \quad \mathbf{L} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} \frac{AB^2}{4} & 0 \\ 0 & \frac{th_s^2}{4} \end{pmatrix}$$

We will therefore consider that the segment $[AB]$ corresponds to a random vector $\underline{S}_{AB}$ following a normal distribution defined by :

$$\underline{S}_{AB} \sim \mathcal{N}\left(\bar{S}_{AB}, \mathbf{C}_{AB}\right) \quad \text{with} \quad \bar{S}_{AB} = \begin{pmatrix} (x_A + x_B)/2 \\ (y_A + y_B)/2 \end{pmatrix} \quad \text{and} \quad \mathbf{C}_{AB} = \mathbf{VLV}^{-1}$$

After calculation, the matrix $\mathbf{C}_{AB}$ est defined as follows:

$$\mathbf{C}_{AB} = \begin{pmatrix} \frac{e_s^2 {y_{AB}}^2}{4\left(y_{AB}^2 + x_{AB}^2\right)} + \frac{{x_{AB}}^2}{4} & \frac{x_{AB}y_{AB}}{4} - \frac{{e_s}^2 x_{AB}y_{AB}}{4\left(y_{AB} + {x_{AB}}^2\right)} \\ \frac{x_{AB}y_{AB}}{4} - \frac{{e_s}^2 x_{AB}y_{AB}}{4\left(y_{AB} + x_{AB}\right)} & \frac{{e_s}^2 {x_{AB}}^2}{4\left(y_{AB}^2 + {x_{AB}}^2\right)} + \frac{{y_{AB}}^2}{4} \end{pmatrix}$$

Now that we represent the segment $[AB]$ with a random vector, we are looking for the probability P_{AB} of intersection between the segment and the probable area of presence of the robot. It is therefore necessary to calculate the following relation:

$$P_{AB} = \int \mathcal{N}\left(\underline{\bar{X}}, \mathbf{C}_x\right) \mathcal{N}\left(\bar{S}_{AB}, \mathbf{C}_{AB}\right) d\underline{X} \tag{10.21}$$

The product of two Gaussians is another Gaussian defined as follows:

$$\mathcal{N}\left(\underline{X}, \mathbf{C}_x\right) \mathcal{N}\left(\bar{S}_{AB}, \mathbf{C}_{AB}\right) = k\mathcal{N}(\underline{\mu}, \mathbf{C})$$

This new Gaussian will be denormalized by the parameter k. It can be written as follows:

$$P_{AB} = \int \mathcal{N}\left(\underline{\bar{X}}, \mathbf{C}_x\right) \mathcal{N}\left(\bar{S}_{AB}, \mathbf{C}_{AB}\right) d\underline{X} = \int k\mathcal{N}(\underline{\mu}, \mathbf{C})\underline{d} = k$$

In order to compute this probability we will use the canonical representation of Gaussians as presented by Murphy *et al.*[20]. Lets us consider a probability density function of parameter $\underline{x}$ (of dimension n) $\mathcal{N}(\underline{x}, \underline{\mu}, C)$ We can then define its canonical representation as follows:

$$\mathcal{N}(\underline{\mu}, C) = \phi(g, h, K) \tag{10.22}$$

$$= \exp(g + \underline{x}^T \mu - \frac{1}{2}\underline{x}^T K x) \tag{10.23}$$

- $K = C^{-1}$
- $h = C^{-1}\mu$
- $g = \log(p) - \frac{1}{2}\underline{\mu}^T K\underline{\mu}$
- $p = \log[(2\pi)^{-n/2}|C|^{-1/2}]$

To find the parameters of the Gaussian from the canonical form, we use the following formulas:

$$\phi(g', \underline{h}, K) = k\mathcal{N}(\underline{\mu}, C) \tag{10.24}$$

with :

- $C = K^{-1}$
- $\underline{\mu} = C.h$
- $k = \frac{1}{p}\exp(g' + \frac{1}{2}\underline{\mu}^T K\underline{\mu})$
- $p = \log[(2\pi)^{-n/2}|C|^{-1/2}]$
- $g' = log(kp) - \frac{1}{2}\underline{\mu}^T K\mu$

This canonical representation simplifies the product of two Gaussians $\mathcal{N}(\mu_1, C_1)$ and $\mathcal{N}(\mu_2, C_2)$:

$$\phi\left(g_1, \underline{h}_1, \mathbf{K}_1\right) \times \phi\left(g_2, \underline{h}_2, \mathbf{K}_2\right) = \phi\left(g_1 + g_2, \underline{h}_1 + \underline{h}_2, \mathbf{K}_1 + \mathbf{K}_2\right)$$

with

$$\mathbf{K}_1 = \mathbf{C}_1^{-1}, \underline{h}_1 = \mathbf{C}_1^{-1}\mu_1, \mathbf{K}_2 = \mathbf{C}_2^{-1} \text{ et } \underline{h}_2 = \mathbf{C}_2^{-1}\mu_2$$

Hence, the product of two Gaussians $\mathcal{N}(\mu_1, C_1)$ and $\mathcal{N}(\mu_2, C_2)$ will be a non-normalized Gaussian:

$$\mathcal{N}\left(\underline{\mu}_1, \mathbf{C}_1\right) \times \mathcal{N}\left(\underline{\mu}_2, \mathbf{C}_2\right) = k\mathcal{N}(\underline{\mu}, \mathbf{C})$$

with

$$\mathbf{C} = \left[\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right]^{-1}$$
$$\mu = \left[\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right]^{-1}\left[\mathbf{C}_1^{-1}\underline{\mu}_1 + \mathbf{C}_2^{-1}\underline{\mu}_2\right]$$

and the parameter k is defined as follows

$$k = \frac{1}{p}\exp\left(g_1 + g_2 + \frac{1}{2}\underline{\mu}^T\mathbf{C}^{-1}\underline{\mu}\right)$$

$$\begin{cases} p = \log\left[(2\pi)^{-n/2}|\mathbf{C}|^{-1/2}\right] \\ g_1 = \log(p_1) - \frac{1}{2}\mu_1^T\mathbf{C}_1^{-1}\underline{\mu}_1 \\ p_1 = \log\left[(2\pi)^{-n/2}\left|\mathbf{C}_1\right|^{-1/2}\right. \\ g_2 = \log(p_2) - \frac{1}{2}\underline{\mu}_2^T\mathbf{C}_2^{-1}\underline{\mu}_2 \\ p_2 = \log\left[(2\pi)^{-n/2}\left|\mathbf{C}_2\right|^{-1/2}\right] \end{cases}$$

Going back to our problematic, the probability P_{AB} is then defined as follows:

$$P_{AB} = k = \frac{1}{p} \exp\left(g_1 + g_2 + \frac{1}{2}\underline{\mu}^T \mathbf{C}^{-1} \underline{\mu}\right) \tag{10.25}$$

with:

- $\mathbf{C} = \left[\mathbf{C}_X^{-1} + \mathbf{C}_{AB}^{-1}\right]^{-1}$
- $\underline{\mu} = \left[\mathbf{C}_X^{-1} + \mathbf{C}_{AB}^{-1}\right]^{-1} \left[\mathbf{C}_X^{-1} \underline{X} + \mathbf{C}_{AB}^{-1} \bar{S}_{AB}\right]$
- $p = \log\left[(2\pi)^{-n/2} |\mathbf{C}|^{-1/2}\right]$
- $g_1 = \log(2\pi)^{n/2} \left|\mathbf{C}_X\right|^{-1/2} - \frac{1}{2}\underline{X}^1 \mathbf{C}_X^{-1} \underline{\bar{X}}$
- $g_2 = \log\left[(2\pi)^{n/2} \left|\mathbf{C}_{AB}\right|^{-1/2}\right] - \frac{1}{2}\bar{S}_{AB}^T \mathbf{C}_{AB}^{-1} \bar{S}_{AB}$

BIBLIOGRAPHY

[1] FRÉCHET, M. M. **Sur quelques points du calcul fonctionnel**. *Rendiconti del Circolo Matematico di Palermo (1884-1940) 22*, 1 (1906), 1–72.

[2] MCCULLOCH, W. S., AND PITTS, W. **A logical calculus of the ideas immanent in nervous activity**. *The bulletin of mathematical biophysics 5*, 4 (1943), 115–133.

[3] VITERBI, A. **Error bounds for convolutional codes and an asymptotically optimum decoding algorithm**. *IEEE transactions on Information Theory 13*, 2 (1967), 260–269.

[4] FENTON, R. E. **Automatic vehicle guidance and control—a state of the art survey**. *IEEE Transactions on Vehicular Technology 19*, 1 (1970), 153–161.

[5] REID, D. **An algorithm for tracking multiple targets**. *IEEE transactions on Automatic Control 24*, 6 (1979), 843–854.

[6] TSUGAWA, S., YATABE, T., HIROSE, T., AND MATSUMOTO, S. **An automobile with artificial intelligence**. In *Proceedings of the 6th international joint conference on Artificial intelligence-Volume 2* (1979), pp. 893–895.

[7] DICKMANNS, E. D., AND ZAPP, A. **A curvature-based scheme for improving road vehicle guidance by computer vision**. In *Mobile Robots I* (1987), vol. 727, International Society for Optics and Photonics, pp. 161–168.

[8] DICKMANNS, E. D., AND GRAEFE, V. **Dynamic monocular machine vision**. *Machine vision and applications 1*, 4 (1988), 223–240.

[9] MCMASTER, R. B., AND SHEA, K. S. **Generalization in digital cartography**. Association of American Geographers Washington, DC.

[10] TSUGAWA, S. **Vision-based vehicles in japan: Machine vision systems and driving control systems**. *IEEE Transactions on industrial electronics 41*, 4 (1994), 398–405.

[11] POMERLEAU, D. **Ralph: Rapidly adapting lateral position handler**. In *Proceedings of the Intelligent Vehicles' 95. Symposium* (1995), IEEE, pp. 506–511.

[12] BERNSTEIN, D., KORNHAUSER, A., AND OTHERS. **An introduction to map matching for personal navigation assistants**. New Jersey TIDE Center, 1996.

[13] DICKMANNS, E. D., AND OTHERS. **Vehicles capable of dynamic vision**. In *IJCAI* (1997), vol. 97, pp. 1577–1592.

[14] MITSUBISHI MOTORS. **Mitsubishi motors develops new driver support system**, December 1998. Consulted on december 2020: https://www.mitsubishi-motors.com/en/corporate/pressrelease/corporate/detail429.html.

[15] DMITRIEV, S., STEPANOV, A., RIVKIN, B., KOSHAEV, D., AND CHUNG, D. **Optimal map-matching for car navigation systems**. In *Proc. of 6th International Conference on Integrated Navigation Systems, St. Petersburg* (1999).

[16] GONZALEZ, J. P., AND OZGUNER, U. **Lane detection using histogram-based segmentation and decision trees**. In *ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 00TH8493)* (2000), IEEE, pp. 346–351.

[17] WHITE, C. E., BERNSTEIN, D., AND KORNHAUSER, A. L. **Some map matching algorithms for personal navigation assistants**. *Transportation research part c: emerging technologies 8*, 1-6 (2000), 91–108.

[18] AUFRERE, R., CHAPUIS, R., AND CHAUSSE, F. **A model-driven approach for real-time road recognition**. *Machine Vision and Applications 13*, 2 (2001), 95–107.

[19] PYO, J.-S., SHIN, D.-H., AND SUNG, T.-K. **Development of a map matching method using the multiple hypothesis technique**. In *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585)* (2001), IEEE, pp. 23–27.

[20] MURPHY, K. P. **Dynamic bayesian networks: representation, inference and learning**. PhD thesis, University of California, Berkeley Berkeley, CA, 2002.

[21] ALT, H., EFRAT, A., ROTE, G., AND WENK, C. **Matching planar maps**. *Journal of algorithms 49*, 2 (2003), 262–283.

[22] SCHWARTZ, D. A. **Clothoid road geometry unsuitable for sensor fusion clothoid parameter sloshing**. In *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No. 03TH8683)* (2003), IEEE, pp. 484–488.

[23] YIM, Y. U., AND OH, S.-Y. **Three-feature based automatic lane detection algorithm (tfalda) for autonomous driving**. *IEEE Transactions on Intelligent Transportation Systems 4*, 4 (2003), 219–225.

[24] URMSON, C., ANHALT, J., CLARK, M., GALATALI, T., GONZALEZ, J. P., GOWDY, J., GUTIERREZ, A., HARBAUGH, S., JOHNSON-ROBERSON, M., KATO, H., AND OTHERS. **High speed navigation of unrehearsed terrain: Red team technology for grand challenge 2004**. *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-04-37* (2004).

[25] BRAKATSOULAS, S., PFOSER, D., SALAS, R., AND WENK, C. **On map-matching vehicle tracking data**. In *Proceedings of the 31st international conference on Very large data bases* (2005), pp. 853–864.

[26] MARCHAL, F., HACKNEY, J., AND AXHAUSEN, K. W. **Efficient map matching of large global positioning system data sets: Tests on speed-monitoring experiment in zürich**. *Transportation Research Record 1935*, 1 (2005), 93–100.

[27] RASMUSSEN, C., AND KORAH, T. **On-vehicle and aerial texture analysis for vision-based desert road following**. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops* (2005), IEEE, pp. 66–66.

[28] ALON, Y., FERENCZ, A., AND SHASHUA, A. **Off-road path following using region classification and geometric projection constraints**. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)* (2006), vol. 1, IEEE, pp. 689–696.

[29] CHENG, H.-Y., JENG, B.-S., TSENG, P.-T., AND FAN, K.-C. **Lane detection with moving vehicles in the traffic scenes**. *IEEE Transactions on intelligent transportation systems 7*, 4 (2006), 571–582.

[30] HUMMEL, B. **Map matching for vehicle guidance**. In *Dynamic and Mobile GIS*. CRC Press, 2006, pp. 196–207.

[31] MCCALL, J. C., AND TRIVEDI, M. M. **Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation**. *IEEE transactions on intelligent transportation systems 7*, 1 (2006), 20–37.

[32] OGAWA, T., AND TAKAGI, K. **Lane recognition using on-vehicle lidar**. In *2006 IEEE Intelligent Vehicles Symposium* (2006), IEEE, pp. 540–545.

[33] QUDDUS, M. A. **High integrity map matching algorithms for advanced transport telematics applications**. PhD thesis, Imperial College London London, 2006.

[34] SAMADZADEGAN, F., SARAFRAZ, A., AND TABIBI, M. **Automatic lane detection in image sequences for vision-based navigation purposes**. *ISPRS Image Engineering and Vision Metrology* (2006).

[35] SAWANO, H., AND OKADA, M. **A road extraction method by an active contour model with inertia and differential features**. *IEICE transactions on information and systems 89*, 7 (2006), 2257–2267.

[36] THRUN, S., MONTEMERLO, M., DAHLKAMP, H., STAVENS, D., ARON, A., DIEBEL, J., FONG, P., GALE, J., HALPENNY, M., HOFFMANN, G., AND OTHERS. **Stanley: The robot that won the darpa grand challenge**. *Journal of field Robotics 23*, 9 (2006), 661–692.

[37] WENK, C., SALAS, R., AND PFOSER, D. **Addressing the need for map-matching speed: Localizing global curve-matching algorithms**. In *18th international conference on scientific and statistical database management (SSDBM'06)* (2006), IEEE, pp. 379–388.

[38] ÁLVAREZ, J. M., LÓPEZ, A. M., AND BALDRICH, R. **Shadow resistant road segmentation from a mobile monocular system**. In *Iberian Conference on Pattern Recognition and Image Analysis* (2007), Springer, pp. 9–16.

[39] QUDDUS, M. A., OCHIENG, W. Y., AND NOLAND, R. B. **Current map-matching algorithms for transport applications: State-of-the art and future research directions**. *Transportation research part c: Emerging technologies 15*, 5 (2007), 312–328.

[40] ALY, M. **Real time detection of lane markers in urban streets**. In *2008 IEEE Intelligent Vehicles Symposium* (2008), IEEE, pp. 7–12.

[41] BACHA, A., BAUMAN, C., FARUQUE, R., FLEMING, M., TERWELP, C., REINHOLTZ, C., HONG, D., WICKS, A., ALBERI, T., ANDERSON, D., AND OTHERS. **Team victortango's entry in the darpa urban challenge, vol. 25, no. 1**. *June 10* (2008), 467–492.

[42] DU, J., AND BARTH, M. J. **Next-generation automated vehicle location systems: Positioning at the lane level**. *IEEE Transactions on Intelligent Transportation Systems 9*, 1 (2008), 48–57.

[43] HAKLAY, M., AND WEBER, P. **Openstreetmap: User-generated street maps**. *IEEE Pervasive Computing 7*, 4 (2008), 12–18.

[44] JABBOUR, M., BONNIFAIT, P., AND CHERFAOUI, V. **Map-matching integrity using multihypothesis road-tracking**. *Journal of Intelligent Transportation Systems 12*, 4 (2008), 189–201.

[45] KIM, Z. **Robust lane detection and tracking in challenging scenarios**. *IEEE Transactions on Intelligent Transportation Systems 9*, 1 (2008), 16–26.

[46] LIPSKI, C., SCHOLZ, B., BERGER, K., LINZ, C., STICH, T., AND MAGNOR, M. **A fast and robust approach to lane marking detection and lane tracking**. In *2008 ieee southwest symposium on image analysis and interpretation* (2008), IEEE, pp. 57–60.

[47] NIETO, M., SALGADO, L., JAUREGUIZAR, F., AND ARRÓSPIDE, J. **Robust multiple lane road modeling based on perspective analysis**. In *2008 15th IEEE International Conference on Image Processing* (2008), IEEE, pp. 2396–2399.

[48] PINK, O., AND HUMMEL, B. **A statistical approach to map matching using road network geometry, topology and vehicular motion constraints**. In *2008 11th International IEEE conference on intelligent transportation systems* (2008), IEEE, pp. 862–867.

[49] VEIT, T., TAREL, J.-P., NICOLLE, P., AND CHARBONNIER, P. **Evaluation of road marking feature extraction**. In *2008 11th International IEEE Conference on Intelligent Transportation Systems* (2008), IEEE, pp. 174–181.

[50] WU, S.-J., CHIANG, H.-H., PERNG, J.-W., CHEN, C.-J., WU, B.-F., AND LEE, T.-T. **The heterogeneous systems integration design and implementation for lane keeping on a vehicle**. *IEEE Transactions on Intelligent Transportation Systems 9*, 2 (2008), 246–263.

[51] YAMAGUCHI, K., WATANABE, A., NAITO, T., AND NINOMIYA, Y. **Road region estimation using a sequence of monocular images**. In *2008 19th International Conference on Pattern Recognition* (2008), IEEE, pp. 1–4.

[52] BORKAR, A., HAYES, M., AND SMITH, M. T. **Robust lane detection and tracking with ransac and kalman filter**. In *2009 16th IEEE International Conference on Image Processing (ICIP)* (2009), IEEE, pp. 3261–3264.

[53] HERNÁNDEZ, J., AND MARCOTEGUI, B. **Filtering of artifacts and pavement segmentation from mobile lidar data**. In *ISPRS Workshop Laserscanning 2009* (2009).

[54] HUANG, A. S., MOORE, D., ANTONE, M., OLSON, E., AND TELLER, S. **Finding multiple lanes in urban road networks with vision and lidar**. *Autonomous Robots 26*, 2 (2009), 103–122.

[55] JIANG, R., KLETTE, R., VAUDREY, T., AND WANG, S. **New lane model and distance transform for lane detection and tracking**. In *International Conference on Computer Analysis of Images and Patterns* (2009), Springer, pp. 1044–1052.

[56] KATRAMADOS, I., CRUMPLER, S., AND BRECKON, T. P. **Real-time traversable surface detection by colour space fusion and temporal analysis**. In *International Conference on Computer Vision Systems* (2009), Springer, pp. 265–274.

[57] KONG, H., AUDIBERT, J.-Y., AND PONCE, J. **Vanishing point detection for road detection**. In *2009 ieee conference on computer vision and pattern recognition* (2009), IEEE, pp. 96–103.

[58] LOU, Y., ZHANG, C., ZHENG, Y., XIE, X., WANG, W., AND HUANG, Y. **Map-matching for low-sampling-rate gps trajectories**. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), pp. 352–361.

[59] NEWSON, P., AND KRUMM, J. **Hidden markov map matching through noise and sparseness**. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), pp. 336–343.

[60] TOLEDO-MOREO, R., BÉTAILLE, D., AND PEYRET, F. **Lane-level integrity provision for navigation and map matching with gnss, dead reckoning, and enhanced maps**. *IEEE Transactions on Intelligent Transportation Systems 11*, 1 (2009), 100–112.

[61] ZHANG, G., ZHENG, N., CUI, C., YAN, Y., AND YUAN, Z. **An efficient road detection method in noisy urban environment**. In *2009 IEEE Intelligent Vehicles Symposium* (2009), IEEE, pp. 556–561.

[62] BENNETT, J. **OpenStreetMap**. Packt Publishing Ltd, 2010.

[63] GACKSTATTER, C., HEINEMANN, P., THOMAS, S., AND KLINKER, G. **Stable road lane model based on clothoids**. In *Advanced Microsystems for Automotive Applications 2010*. Springer, 2010, pp. 133–143.

[64] LIU, G., WÖRGÖTTER, F., AND MARKELIĆ, I. **Combining statistical hough transform and particle filter for robust lane detection and tracking**. In *2010 IEEE Intelligent Vehicles Symposium* (2010), IEEE, pp. 993–997.

[65] LOOSE, H., AND FRANKE, U. **B-spline-based road model for 3d lane recognition**. In *13th International IEEE Conference on Intelligent Transportation Systems* (2010), IEEE, pp. 91–98.

[66] RUSSELL, S. J., AND NORVIG, P. **Artificial intelligence-a modern approach (3rd internat. edn.)**, 2010.

[67] VELAGA, N. R. **Development of a weight-based topological map-matching algorithm and an integrity method for location-based ITS services**. PhD thesis, © Nagendra R. Velaga, 2010.

[68] CHEN, D., DRIEMEL, A., GUIBAS, L. J., NGUYEN, A., AND WENK, C. **Approximate map matching with respect to the fréchet distance**. In *2011 Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments (ALENEX)* (2011), SIAM, pp. 75–83.

[69] POPESCU, V., BACE, M., AND NEDEVSCHI, S. **Lane identification and ego-vehicle accurate global positioning in intersections**. In *2011 IEEE Intelligent Vehicles Symposium (IV)* (2011), IEEE, pp. 870–875.

[70] GEIGER, A., LENZ, P., AND URTASUN, R. **Are we ready for autonomous driving? the kitti vision benchmark suite**. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), IEEE, pp. 3354–3361.

[71] GOPALAN, R., HONG, T., SHNEIER, M., AND CHELLAPPA, R. **A learning approach towards detection and tracking of lane markings**. *IEEE Transactions on Intelligent Transportation Systems 13*, 3 (2012), 1088–1098.

[72] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. **Imagenet classification with deep convolutional neural networks**. *Advances in neural information processing systems 25* (2012), 1097–1105.

[73] MILANÉS, V., LLORCA, D. F., VILLAGRÁ, J., PÉREZ, J., FERNÁNDEZ, C., PARRA, I., GONZÁLEZ, C., AND SOTELO, M. A. **Intelligent automatic overtaking system using vision for vehicle detection**. *Expert Systems with Applications 39*, 3 (2012), 3362–3373.

[74] POPESCU, V., DANESCU, R., AND NEDEVSCHI, S. **On-road position estimation by probabilistic integration of visual cues**. In *2012 IEEE Intelligent Vehicles Symposium* (2012), IEEE, pp. 583–589.

[75] WU, T., AND RANGANATHAN, A. **A practical system for road marking detection and recognition**. In *2012 IEEE Intelligent Vehicles Symposium* (2012), IEEE, pp. 25–30.

[76] ZHENG, K., ZHENG, Y., XIE, X., AND ZHOU, X. **Reducing uncertainty of low-sampling-rate trajectories**. In *2012 IEEE 28th International Conference on Data Engineering* (2012), IEEE, pp. 1144–1155.

[77] GEIGER, A., LENZ, P., STILLER, C., AND URTASUN, R. **Vision meets robotics: The kitti dataset**. *The International Journal of Robotics Research 32*, 11 (2013), 1231–1237.

[78] HUNTER, T., ABBEEL, P., AND BAYEN, A. **The path inference filter: model-based low-latency map matching of probe vehicle data**. *IEEE Transactions on Intelligent Transportation Systems 15*, 2 (2013), 507–529.

[79] LI, L., QUDDUS, M., AND ZHAO, L. **High accuracy tightly-coupled integrity monitoring algorithm for map-matching**. *Transportation Research Part C: Emerging Technologies 36* (2013), 13–26.

[80] SERMANET, P., EIGEN, D., ZHANG, X., MATHIEU, M., FERGUS, R., AND LECUN, Y. **Overfeat: Integrated recognition, localization and detection using convolutional networks**. *arXiv preprint arXiv:1312.6229* (2013).

[81] SZEGEDY, C., ZAREMBA, W., SUTSKEVER, I., BRUNA, J., ERHAN, D., GOODFELLOW, I., AND FERGUS, R. **Intriguing properties of neural networks**. *arXiv preprint arXiv:1312.6199* (2013).

[82] WEI, H., WANG, Y., FORMAN, G., AND ZHU, Y. **Map matching by fréchet distance and global weight optimization**. *Technical Paper, Departement of Computer Science and Engineering* (2013), 19.

[83] ALVAREZ, J. M., LOPEZ, A. M., GEVERS, T., AND LUMBRERAS, F. **Combining priors, appearance, and context for road detection**. *IEEE Transactions on Intelligent Transportation Systems 15*, 3 (2014), 1168–1178.

[84] AYNAUD, C., BERNAY-ANGELETTI, C., CHAPUIS, R., AUFRÈRE, R., DEBAIN, C., AND KARAM, N. **Real-time vehicle localization by using a top-down process**. In *17th International Conference on Information Fusion (FUSION)* (2014), IEEE, pp. 1–6.

[85] BENDER, P., ZIEGLER, J., AND STILLER, C. **Lanelets: Efficient map representation for autonomous driving**. In *2014 IEEE Intelligent Vehicles Symposium Proceedings* (2014), IEEE, pp. 420–425.

[86] COMMITTEE, S. O.-R. A. V. S. **Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems**. *SAE Standard J 3016* (2014), 1–16.

[87] HATA, A., AND WOLF, D. **Road marking detection using LIDAR reflective intensity data and its application to vehicle localization**. *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014* (2014), 584–589.

[88] HATA, A., AND WOLF, D. **Road marking detection using lidar reflective intensity data and its application to vehicle localization**. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (2014), IEEE, pp. 584–589.

[89] HILLEL, A. B., LERNER, R., LEVI, D., AND RAZ, G. **Recent progress in road and lane detection: a survey**. *Machine vision and applications 25*, 3 (2014), 727–745.

[90] KUBIČKA, M., CELA, A., MOUNIER, H., AND NICULESCU, S.-I. **On designing robust real-time map-matching algorithms**. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (2014), IEEE, pp. 464–470.

[91] LU, W., SEIGNEZ, E., RODRIGUEZ, F. S. A., AND REYNAUD, R. **Lane marking based vehicle localization using particle filter and multi-kernel estimation**. In *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)* (2014), IEEE, pp. 601–606.

[92] SHIN, B.-S., XU, Z., AND KLETTE, R. **Visual lane analysis and higher-order tasks: a concise review**. *Machine vision and applications 25*, 6 (2014), 1519–1547.

[93] TEAM, G. P. **Global positioning system (gps) standard positioning service (sps) performance analysis report**. Tech. rep., Washington: Federal Aviation Administration, 2014.

[94] AYNAUD, C. **Localisation précise et fiable de véhicules par approche multisensorielle**. PhD thesis, Université Blaise Pascal-Clermont-Ferrand II, 2015.

[95] CUI, D., XUE, J., AND ZHENG, N. **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 (2015), 1039–1050.

[96] HUVAL, B., WANG, T., TANDON, S., KISKE, J., SONG, W., PAZHAYAMPALLIL, J., ANDRILUKA, M., RAJPURKAR, P., MIGIMATSU, T., CHENG-YUE, R., AND OTHERS. **An empirical evaluation of deep learning on highway driving**. *arXiv preprint arXiv:1504.01716* (2015).

[97] RONNEBERGER, O., FISCHER, P., AND BROX, T. **U-net: Convolutional networks for biomedical image segmentation**. In *International Conference on Medical image computing and computer-assisted intervention* (2015), Springer, pp. 234–241.

[98] YANG, J., AND MENG, L. **Feature selection in conditional random fields for map matching of gps trajectories**. In *Progress in Location-Based Services 2014*. Springer, 2015, pp. 121–135.

[99] BAUER, S., ALKHORSHID, Y., AND WANIELIK, G. **Using high-definition maps for precise urban vehicle localization**. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* (2016), IEEE, pp. 492–497.

[100] CÁCERES HERNÁNDEZ, D., KURNIANGGORO, L., FILONENKO, A., AND JO, K. H. **Real-time lane region detection using a combination of geometrical and image features**. *Sensors 16*, 11 (2016), 1935.

[101] GURGHIAN, A., KODURI, T., BAILUR, S. V., CAREY, K. J., AND MURALI, V. N. **Deeplanes: End-to-end lane position estimation using deep neural networksa**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2016), pp. 38–45.

[102] HE, B., AI, R., YAN, Y., AND LANG, X. **Accurate and robust lane detection based on dual-view convolutional neutral network**. In *2016 IEEE Intelligent Vehicles Symposium (IV)* (2016), IEEE, pp. 1041–1046.

[103] HU, G., SHAO, J., LIU, F., WANG, Y., AND SHEN, H. T. **If-matching: Towards accurate map-matching with information fusion**. *IEEE Transactions on Knowledge and Data Engineering 29*, 1 (2016), 114–127.

[104] REDMON, J., DIVVALA, S., GIRSHICK, R., AND FARHADI, A. **You only look once: Unified, real-time object detection**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 779–788.

[105] SUHR, J. K., JANG, J., MIN, D., AND JUNG, H. G. **Sensor fusion-based low-cost vehicle localization system for complex urban environments**. *IEEE Transactions on Intelligent Transportation Systems 18*, 5 (2016), 1078–1086.

[106] SVENSSON, D., AND SÖRSTEDT, J. **Ego lane estimation using vehicle observations and map information**. In *2016 IEEE Intelligent Vehicles Symposium (IV)* (2016), IEEE, pp. 909–914.

[107] WANG, X., AND NI, W. **An improved particle filter and its application to an ins/gps integrated navigation system in a serious noisy scenario**. *Measurement Science and Technology 27*, 9 (2016), 095005.

[108] ABRAMOV, A., BAYER, C., HELLER, C., AND LOY, C. **A flexible modeling approach for robust multi-lane road estimation**. In *2017 IEEE Intelligent Vehicles Symposium (IV)* (2017), IEEE, pp. 1386–1392.

[109] BALLARDINI, A. L., CATTANEO, D., IZQUIERDO, R., PARRA, I., SOTELO, M., AND SORRENTI, D. G. **Ego-lane estimation by modeling lanes and sensor failures**. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (2017), IEEE, pp. 1–7.

[110] BERRIEL, R. F., DE AGUIAR, E., DE SOUZA, A. F., AND OLIVEIRA-SANTOS, T. **Ego-lane analysis system (elas): Dataset and algorithms**. *Image and Vision Computing 68* (2017), 64–75.

[111] GRANTER, S. R., BECK, A. H., AND PAPKE JR, D. J. **Alphago, deep learning, and the future of the human microscopist**. *Archives of pathology & laboratory medicine 141*, 5 (2017), 619–621.

[112] JAGADEESH, G. R., AND SRIKANTHAN, T. **Online map-matching of noisy and sparse location data with hidden markov and route choice models**. *IEEE Transactions on Intelligent Transportation Systems 18*, 9 (2017), 2423–2434.

[113] LEE, S., KIM, J., SHIN YOON, J., SHIN, S., BAILO, O., KIM, N., LEE, T.-H., SEOK HONG, H., HAN, S.-H., AND SO KWEON, I. **Vpgnet: Vanishing point guided network for lane and road marking detection and recognition**. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 1947–1955.

[114] LI, F., BONNIFAIT, P., AND IBANEZ-GUZMAN, J. **Estimating localization uncertainty using multi-hypothesis map-matching on high-definition road maps**. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (2017), IEEE, pp. 1–6.

[115] LI, F., BONNIFAIT, P., AND IBAÑEZ-GUZMÁN, J. **Using high definition maps to estimate gnss positioning uncertainty**. In *European Navigation Conference (ENC 2017)* (2017).

[116] LIU, X., DENG, Z., LU, H., AND CAO, L. **Benchmark for road marking detection: Dataset specification and performance baseline**. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (2017), IEEE, pp. 1–6.

[117] LUO, A., CHEN, S., AND XV, B. **Enhanced map-matching algorithm with a hidden markov model for mobile phone positioning**. *ISPRS International Journal of Geo-Information 6*, 11 (2017), 327.

[118] THEDRIVE. **The untold history of the first driverless car crash—part 1**, January 2017. Consulted on december 2020: https://www.thedrive.com/vintage/6797/the-untold-history-of-the-first-driverless-car-crash-part-1.

[119] XIAO, J., LUO, L., YAO, Y., ZOU, W., AND KLETTE, R. **Lane detection based on road module and extended kalman filter**. In *Pacific-Rim Symposium on Image and Video Technology* (2017), Springer, pp. 382–395.

[120] ZHAO, Y., LI, J., AND YU, L. **A deep learning ensemble approach for crude oil price forecasting**. *Energy Economics 66* (2017), 9–16.

[121] ZHU, H., YUEN, K.-V., MIHAYLOVA, L., AND LEUNG, H. **Overview of environment perception for intelligent vehicles**. *IEEE Transactions on Intelligent Transportation Systems 18*, 10 (2017), 2584–2601.

[122] **Tusimple benchmark. 2018.** https://github.com/TuSimple/tusimple-benchmark, 2018. Accessed: 2021-03-17.

[123] BANSAL, M., KRIZHEVSKY, A., AND OGALE, A. **Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst**. *arXiv preprint arXiv:1812.03079* (2018).

[124] DE PAULA VERONESE, L., ISMAIL, A., NARAYAN, V., AND SCHULZE, M. **An accurate and computational efficient system for detecting and classifying ego and sides lanes using lidar**. In *2018 IEEE Intelligent Vehicles Symposium (IV)* (2018), IEEE, pp. 1476–1483.

[125] DELOBEL, L., AUFRÈRE, R., DEBAIN, C., CHAPUIS, R., AND CHATEAU, T. **A real-time map refinement method using a multi-sensor localization framework**. *IEEE Transactions on Intelligent Transportation Systems 20*, 5 (2018), 1644–1658.

[126] GHAFOORIAN, M., NUGTEREN, C., BAKA, N., BOOIJ, O., AND HOFMANN, M. **El-gan: Embedding loss driven generative adversarial networks for lane detection**. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops* (2018), pp. 0–0.

[127] GHALLABI, F., NASHASHIBI, F., EL-HAJ-SHHADE, G., AND MITTET, M.-A. **Lidar-based lane marking detection for vehicle positioning in an hd map**. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 2209–2214.

[128] HSU, Y.-C., XU, Z., KIRA, Z., AND HUANG, J. **Learning to cluster for proposal-free instance segmentation**. In *2018 International Joint Conference on Neural Networks (IJCNN)* (2018), IEEE, pp. 1–8.

[129] HUANG, X., CHENG, X., GENG, Q., CAO, B., ZHOU, D., WANG, P., LIN, Y., AND YANG, R. **The apolloscape dataset for autonomous driving**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2018), pp. 954–960.

[130] IBERRAKEN, D., ADOUANE, L., AND DENIS, D. **Multi-level bayesian decision-making for safe and flexible autonomous navigation in highway environment**. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), IEEE, pp. 3984–3990.

[131] IBERRAKEN, D., ADOUANE, L., AND DENIS, D. **Safe autonomous overtaking maneuver based on inter-vehicular distance prediction and multi-level bayesian decision-making**. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 3259–3265.

[132] JANG, E. S., SUHR, J. K., AND JUNG, H. G. **Lane endpoint detection and position accuracy evaluation for sensor fusion-based vehicle localization on highways**. *Sensors 18*, 12 (2018), 4389.

[133] KASMI, A., DENIS, D., AUFRÈRE, R., AND CHAPUIS, R. **Map matching and lanes number estimation with openstreetmap**. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 2659–2664.

[134] KUBICKA, M., CELA, A., MOUNIER, H., AND NICULESCU, S.-I. **Comparative study and application-oriented classification of vehicular map-matching methods**. *IEEE Intelligent Transportation Systems Magazine 10*, 2 (2018), 150–166.

[135] LI, X., ZHAO, K., CONG, G., JENSEN, C. S., AND WEI, W. **Deep representation learning for trajectory similarity computation**. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)* (2018), IEEE, pp. 617–628.

[136] NAVA, D., PANZANI, G., ZAMPIERI, P., AND SAVARESI, S. M. **A two-wheeled vehicle oriented lane detection algorithm**. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 423–428.

[137] NEVEN, D., DE BRABANDERE, B., GEORGOULIS, S., PROESMANS, M., AND VAN GOOL, L. **Towards end-to-end lane detection: an instance segmentation approach**. In *2018 IEEE intelligent vehicles symposium (IV)* (2018), IEEE, pp. 286–291.

[138] PAN, X., SHI, J., LUO, P., WANG, X., AND TANG, X. **Spatial as deep: Spatial cnn for traffic scene understanding**. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2018), vol. 32.

[139] XIAO, Z., JIANG, K., XIE, S., WEN, T., YU, C., AND YANG, D. **Monocular vehicle self-localization method based on compact semantic map**. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (2018), IEEE, pp. 3083–3090.

[140] YU, F., XIAN, W., CHEN, Y., LIU, F., LIAO, M., MADHAVAN, V., AND DARRELL, T. **Bdd100k: A diverse driving video database with scalable annotation tooling**. *arXiv preprint arXiv:1805.04687 2*, 5 (2018), 6.

[141] BEHRENDT, K., AND SOUSSAN, R. **Unsupervised labeled lane markers using maps**. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (2019), pp. 832–839.

[142] BURNETT, K., SCHIMPE, A., SAMAVI, S., GRIDSETH, M., LIU, C. W., LI, Q., KROEZE, Z., AND SCHOELLIG, A. P. **Building a winning self-driving car in six months**. In *2019 International Conference on Robotics and Automation (ICRA)* (2019), IEEE, pp. 9583–9589.

[143] GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems**. O'Reilly Media, 2019.

[144] HOU, Y. **Agnostic lane detection**. *arXiv preprint arXiv:1905.03704* (2019).

[145] HOU, Y., MA, Z., LIU, C., AND LOY, C. C. **Learning lightweight lane detection cnns by self attention distillation**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 1013–1021.

[146] KANG, Y., YIN, H., AND BERGER, C. **Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments**. *IEEE Transactions on Intelligent Vehicles 4*, 2 (2019), 171–185.

[147] KASMI, A., DENIS, D., AUFRÈRE, R., AND CHAPUIS, R. **Probabilistic framework for ego-lane determination**. In *2019 IEEE Intelligent Vehicles Symposium (IV)* (2019), IEEE, pp. 1746–1752.

[148] LI, B., SONG, D., RAMCHANDANI, A., CHENG, H.-M., WANG, D., XU, Y., AND CHEN, B. **Virtual lane boundary generation for human-compatible autonomous driving: A tight coupling between perception and planning**. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019), IEEE, pp. 3733–3739.

[149] LI, X., LI, J., HU, X., AND YANG, J. **Line-cnn: End-to-end traffic line detection with line proposal unit**. *IEEE Transactions on Intelligent Transportation Systems 21*, 1 (2019), 248–258.

[150] MA, W.-C., TARTAVULL, I., BÂRSAN, I. A., WANG, S., BAI, M., MATTYUS, G., HOMAYOUNFAR, N., LAKSHMIKANTH, S. K., POKROVSKY, A., AND URTASUN, R. **Exploiting sparse semantic hd maps for self-driving vehicle localization**. *arXiv preprint arXiv:1908.03274* (2019).

[151] PHILION, J. **Fastdraw: Addressing the long tail of lane detection by adapting a sequential prediction network**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 11582–11591.

[152] PIZZATI, F., ALLODI, M., BARRERA, A., AND GARCÍA, F. **Lane detection and classification using cascaded cnns**. In *International Conference on Computer Aided Systems Theory* (2019), Springer, pp. 95–103.

[153] WELTE, A., XU, P., BONNIFAIT, P., AND ZINOUNE, C. **Estimating the reliability of georeferenced lane markings for map-aided localization**. In *2019 IEEE Intelligent Vehicles Symposium (IV)* (2019), IEEE, pp. 1225–1231.

[154] ZHAO, K., FENG, J., XU, Z., XIA, T., CHEN, L., SUN, F., GUO, D., JIN, D., AND LI, Y. **Deepmm: Deep learning based map matching with data augmentation**. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (2019), pp. 452–455.

[155] BADUE, C., GUIDOLINI, R., CARNEIRO, R. V., AZEVEDO, P., CARDOSO, V. B., FORECHI, A., JESUS, L., BERRIEL, R., PAIXAO, T. M., MUTZ, F., AND OTHERS. **Self-driving cars: A survey**. *Expert Systems with Applications* (2020), 113816.

[156] CAESAR, H., BANKITI, V., LANG, A. H., VORA, S., LIONG, V. E., XU, Q., KRISHNAN, A., PAN, Y., BALDAN, G., AND BEIJBOM, O. **nuscenes: A multimodal dataset for autonomous driving**. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2020), pp. 11621–11631.

[157] CHAO, P., XU, Y., HUA, W., AND ZHOU, X. **A survey on map-matching algorithms**. In *Australasian Database Conference* (2020), Springer, pp. 121–133.

[158] CUDRANO, P., MENTASTI, S., MATTEUCCI, M., BERSANI, M., ARRIGONI, S., AND CHELI, F. **Advances in centerline estimation for autonomous lateral control**. In *2020 IEEE Intelligent Vehicles Symposium (IV)* (2020), IEEE, pp. 1415–1422.

[159] FAYYAD, J., JARADAT, M. A., GRUYER, D., AND NAJJARAN, H. **Deep learning sensor fusion for autonomous vehicle perception and localization: A review**. *Sensors 20*, 15 (2020), 4220.

[160] HOU, Y., MA, Z., LIU, C., HUI, T.-W., AND LOY, C. C. **Inter-region affinity distillation for road marking segmentation**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 12486–12495.

[161] JANAI, J., GÜNEY, F., BEHL, A., GEIGER, A., AND OTHERS. **Computer vision for autonomous vehicles: Problems, datasets and state of the art**. *Foundations and Trends® in Computer Graphics and Vision 12*, 1–3 (2020), 1–308.

[162] JUNG, S., CHOI, S., KHAN, M. A., AND CHOO, J. **Towards lightweight lane detection by optimizing spatial embedding**. *arXiv preprint arXiv:2008.08311* (2020).

[163] KANG, J. M., YOON, T. S., KIM, E., AND PARK, J. B. **Lane-level map-matching method for vehicle localization using gps and camera on a high-definition map**. *Sensors 20*, 8 (2020), 2166.

[164] KASMI, A., LACONTE, J., AUFRÈRE, R., DENIS, D., AND CHAPUIS, R. **End-to-end probabilistic ego-vehicle localization framework**. *IEEE Transactions on Intelligent Vehicles* (2020).

[165] KASMI, A., LACONTE, J., AUFRÈRE, R., THEODOSE, R., DENIS, D., AND CHAPUIS, R. **An information driven approach for ego-lane detection using lidar and openstreetmap**. In *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)* (2020), IEEE, pp. 522–528.

[166] KO, Y., JUN, J., KO, D., AND JEON, M. **Key points estimation and point instance segmentation approach for lane detection**. *arXiv preprint arXiv:2002.06604* (2020).

[167] LIANG, D., GUO, Y.-C., ZHANG, S.-K., MU, T.-J., AND HUANG, X. **Lane detection: A survey with new results**. *Journal of Computer Science and Technology 35* (2020), 493–505.

[168] QIN, Z., WANG, H., AND LI, X. **Ultra fast structure-aware deep lane detection**. *arXiv preprint arXiv:2004.11757* (2020).

[169] TABELINI, L., BERRIEL, R., PAIXAO, T. M., BADUE, C., DE SOUZA, A. F., AND OLIVEIRA-SANTOS, T. **Polylanenet: Lane estimation via deep polynomial regression**. *arXiv preprint arXiv:2004.10924* (2020).

[170] TABELINI, L., BERRIEL, R., PAIXÃO, T. M., BADUE, C., DE SOUZA, A. F., AND OLIVERA-SANTOS, T. **Keep your eyes on the lane: Attention-guided lane detection**. *arXiv preprint arXiv:2010.12035* (2020).

[171] XU, H., WANG, S., CAI, X., ZHANG, W., LIANG, X., AND LI, Z. **Curvelane-nas: Unifying lane-sensitive architecture search and adaptive point blending**. In *Computer Vision – ECCV 2020* (Cham, 2020), A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Springer International Publishing, pp. 689–704.

[172] YOO, S., LEE, H. S., MYEONG, H., YUN, S., PARK, H., CHO, J., AND KIM, D. H. **End-to-end lane marker detection via row-wise classification**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2020), pp. 1006–1007.

[173] LEE, S., CHOI, J., AND SEO, S.-W. **Ego-lane index-aware vehicular localisation using the deeproad network for urban environments**. *IET Intelligent Transport Systems* (2021).

Abstract:

Nowadays, Autonomous Vehicles (AVs) are capable of realizing extraordinary and complicated tasks. Notwithstanding these amazing achievements, several challenges arise, one of them is the ability of the autonomous car to perceive its environment in order to properly evaluate the situation with regards to the road environment. Part of this situation evaluation is the knowledge about ego-localization. In the broadest sense, ego-localization is a meaningful concept that can be related to different problematics. However, one interpretation of ego-localization consists of the knowledge of three key components: the road on which the vehicle is traveling (Road Level Localization (RLL)), the ego-lane position (Ego-Lane Level Localization (ELL)), and the lane on which the vehicle is traveling (LLL). Therefore, a reliable ego-localization system has to fulfill the localization's requirement of each of these components.
The objective of this Ph.D. work is to propose a unified, generalized and modular localization system architecture that tackles every aspect of the localization system. In addition that, a focus is given on opensource map OpenSteetMap (OSM) to demonstrate that even a low-cost map can be used to obtain an accurate localization. To do so, an end-to-end framework composed of several interconnected components is presented. This framework is responsible of providing a localization solution on a digital map by developing a robust map-matching algorithm. Furthermore, it permits the localization of the ego-vehicle with respect to ego-lane by proposing a top-down approach that exploits the priors of the map in order to detect the lane marking. Finally, it determines the lane on which the vehicle is traveling by introducing a modular framework that handles the ambiguities in the lane-level localization. The reliability and the flexibility of the overall proposed architecture and its elementary components have been intensively validated, first, individually using different dataset, and secondly, as a whole solution using a collected dataset in the region of Clermont-Ferrand.

Keywords: Autonomous driving, Localization architecture, Probabilistic framework, Lane detection, Fusion Framework.

Résumé :

De nos jours, les véhicules autonomes (VAs) sont capables de réaliser des tâches extraordinaires et compliquées. Malgré ces réalisations étonnantes, plusieurs défis se posent, l'un d'eux étant la capacité de la voiture autonome à percevoir correctement son environnement afin d'évaluer correctement la situation dans laquelle elle se trouve. Cette évaluation de la situation repose en partie sur la connaissance de l'égo-localisation. Au sens large, l'ego-localisation est un concept vaste qui peut être lié à différentes problématiques. Cependant, une interprétation de l'égo-localisation consiste en la connaissance de trois éléments clés : la route sur laquelle le véhicule circule (Road Level Localization (RLL)), la position de du véhciule par rapport aux marquages de sa voie (Ego-Lane Level Localization (ELL)), et la voie sur laquelle le véhicule circule (Lane-Level Localization (LLL)). Par conséquent, un système d'ego-localisation fiable doit répondre aux exigences de localisation de chacun de ces composants. L'objectif de ce travail de doctorat est de proposer une architecture de système de localisation unifiée, généralisée et modulaire qui aborde tous les aspects du système de localisation. En outre, l'accent est mis sur l'utilisation de la carte opensource OpenSteetMap (OSM) pour démontrer que même une carte à faible coût peut être utilisée pour obtenir une localisation précise. Pour ce faire, une architecture probabliste dite end-to-end composé de plusieurs composants interconnectés est présentée. Cette architecture est chargée de fournir une solution de localisation sur une carte numérique en développant un algorithme robuste de correspondance de carte (map-matching). De plus elle permet de localiser l'ego-véhicule par rapport à l'ego-voie en proposant une approche descendante qui exploite les informations a priori de la carte afin de détecter les marquages de l'ego-voie. Enfin, elle permet de déterminer la voie sur laquelle le véhicule se déplace en introduisant une architecture modulaire qui gère les ambiguïtés dans le choix de la bonne voie. La fiabilité et la flexibilité de l'architecture globale proposée et de ses composants élémentaires ont été validées , d'abord individuellement à l'aide de différents ensembles de données, puis en tant que solution globale à l'aide d'un ensemble de données collectées dans la region de Clermont-Ferrand.

Mots-clés : Véhicules autonomes, architecture de localisation, architecture probabiliste, détection des voies, architecture de fusion d'information.