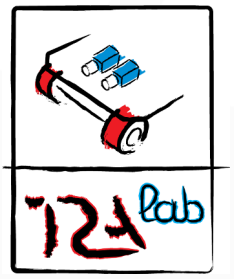
UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

Dipartimento di Informatica Sistemistica e Comunicazione

XXVIII Ph.D. Cycle in Computer Science



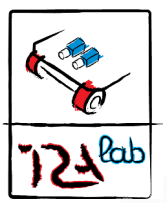
Matching heterogeneous sensing pipelines to digital maps for ego-vehicle localization

PhD dissertation by: **Augusto Luis Ballardini**

Advisor: **Prof. Dr. Domenico G. Sorrenti**

Presented on: **27th March, 2017**

To get what we've never had, we must do what we've never done



Abstract

In this thesis, we present a probabilistic framework for ego-vehicle localization called Road Layout Estimation framework. The main contribution to the vehicle localization problem is the synergistic exploitation of heterogeneous sensing pipelines, as well as their matching with respect to the OpenStreetMap service. The approach is validated in different ways by exploiting different visual clues. Firstly by using the road graph provided by the OpenStreetMap service, then exploiting high-level features like intersections between roads, buildings façades, and other road features. Regarding the effectiveness of the road-graph exploitation, its is proven by achieving real-time computation with state-of-the-art results on a set of ten not trivial runs from the KITTI dataset, including both urban/residential and highway/road scenarios. Moreover, a probabilistic approach for detecting and classifying urban road intersections from a moving vehicle is presented. The approach is based on images from an on-board stereo rig. It relies on the detection of the road ground plane on one side, and on a pixel-level classification of the observed scene on the other. The two processing pipelines are then integrated and the parameters of the road components, i.e., the intersection geometry, are inferred. As opposed to other state-of-the-art off-line methods, which require processing of the whole video sequence up to when the vehicle is inside the intersection, our approach integrates the image data by means of an on-line procedure. The experiments have been performed on the well-known KITTI datasets as well, allowing the community to perform future comparisons. Besides the pure road interpretation schemes, in this work we also present a technique that takes advantage of detected building façades and OpenStreetMaps building data to improve the localization of an autonomous vehicle driving in an urban scenario. The proposed approach also leverages images from the stereo rig mounted on the vehicle to produce a mathematical representation of the buildings' façades within the field of view. This representation is matched against the outlines of the surrounding buildings as they are available on OpenStreetMaps. All the retrieved features are fed into our probabilistic framework, in order to produce an accurate lane-level localization of the vehicle in urban contexts. Finally, as to achieve a lane-level localization also in highway scenarios, we propose two methods that allow the framework to leverage the lane number and the road width. The proposed approaches have been tested under real traffic conditions, showing satisfactory performances with respect to the map-matching-only settings and compensating the noisy measures of a basic line detector.

Contents

1 Introduction **1**
- 1.1 Motivations . . . 1
- 1.2 Problem Statement . . . 2
- 1.3 Challenges . . . 3
- 1.4 Applications . . . 4
- 1.5 Contributions . . . 6
- 1.6 Thesis Outline . . . 7

2 Related Work **9**
- 2.1 Autonomous Driving . . . 9
- 2.2 Localization . . . 12
- 2.3 Grid-Based Maps . . . 13
- 2.4 Feature Based Mapping . . . 15
- 2.5 Topological Mapping . . . 15
- 2.6 3D Scene Understanding . . . 17
- 2.7 From Scene Understanding to Urban Localization . . . 19
- 2.8 Sensing the Environment . . . 21
  - 2.8.1 Buildings . . . 22
  - 2.8.2 Road Intersections . . . 25
  - 2.8.3 Road Features Detection . . . 28
- 2.9 Conclusions . . . 32

3 Ego Vehicle Localization **33**
- 3.1 The Road Layout Estimation Framework . . . 34
  - 3.1.1 Layout Hypotheses . . . 36
  - 3.1.2 Hypotheses Initialization and Evolution . . . 38
  - 3.1.3 Layout components . . . 39
  - 3.1.4 Hypotheses evaluation . . . 39
- 3.2 Leveraging RLE for Vehicle Localization . . . 40
  - 3.2.1 OpenStreetMap features . . . 42
  - 3.2.2 OpenStreetMap module: Hypotheses initialization and Scoring Function . . . 42
- 3.3 The Building Model . . . 43
  - 3.3.1 Façades Detection Pipeline . . . 44

3.3.2 The Buildings Database . . . 46
3.3.3 Detecting the Building Geometry from image data . . . 46
3.3.4 Semantic Segmentation . . . 50
3.3.5 Post Processing . . . 51
3.3.6 Layout Component Scoring function . . . 52
3.4 Road markings: Width and Lanes . . . 58
3.4.1 Line detector . . . 58
3.4.2 Road Width Component . . . 59
3.4.3 Lane Component . . . 63
3.4.4 Final considerations . . . 65
3.5 Conclusions . . . 65

4 Intersection Detector **67**
4.1 The Intersection Model . . . 67
4.2 Geometric Segmentation of the Road . . . 69
4.2.1 Point Cloud Occupancy Grid: PCLOG . . . 70
4.3 Semantic Segmentation . . . 73
4.3.1 Training Dataset . . . 74
4.3.2 CRF Occupancy Grid: CRFOG . . . 76
4.4 Temporal Integration . . . 76
4.5 Increasing the classification consistency . . . 78
4.6 Scoring Function and Classification . . . 80
4.7 Conclusions . . . 82

5 Experimental Evaluation **85**
5.1 OpenStreetMap Matching Pipeline . . . 86
5.1.1 Weak Points . . . 88
5.1.2 Discussion . . . 89
5.2 Buildings Localization Enhancements . . . 91
5.2.1 Experimental results . . . 91
5.2.2 Discussion . . . 96
5.3 Intersection Detection . . . 101
5.3.1 Experimental Results . . . 102
5.3.2 Image Classification Assessment . . . 106
5.3.3 Geometric vs Semantic Intersection Classification . . . 111
5.3.4 Temporal integration . . . 113
5.3.5 Scoring Function Assessment . . . 114
5.3.6 Discussion . . . 114
5.4 Lane and Lines - Highway case study . . . 122
5.4.1 Road Width Component . . . 122
5.4.2 Road Lane Component . . . 124
5.5 Conclusion . . . 130

6 Conclusions and Future Works **133**
6.1 Conclusions . . . 133
6.2 Future Works . . . 134

Bibliography **134**

Chapter 1

Introduction

After approximately one century from the first attempts to create automated driving cars, the recent progress in the context of Intelligent Transportation Systems allows us to consider the autonomous driving cars just a matter of time. Within a relatively short period, human drivers are going to share the road with artificial robotized vehicles, including cars and vehicles of autonomous public transportation systems. Obtaining a high level of reliability for such systems is then mandatory, as to ensure the safety of people's life and realistic usage levels of the vehicles. While active safety systems stemming from the vehicle control theory like the ABS and the ESP are nowadays mandatory, the next generation of Advanced Driver Assistance Systems, or ADAS, will need robust environment perception algorithms in order to securely perform self-driving maneuvers, *i.e.*, dynamic driving tasks on a sustained basis[1]. As every robotic agent, many of these algorithms require an accurate localization within a representation of the environment and a rich description of the surrounding scene in terms of traffic signs, road lanes, other cars, etc. Towards this goal, in this thesis we present a system that allows the vehicles to handle the perceived features of the surrounding scenario and to leverage the information of the cartographic maps retrieved from a mapping provider like the OpenStreetMap service.

1.1 Motivations

Differently from indoor and outdoor mobile robotics platforms, which are generally operated in safe controlled areas, autonomous vehicles are

[1] SAE International J3016 Standard Levels 3 to 5 [1]

required to fulfill common roads regulations in order to safely drive in real-world, human-populated environments. For this reason, these vehicles need a complete situational awareness of their surroundings. On the one hand, such rich representations require perception algorithms aimed at detecting, tracking and avoiding obstacles, as well as to understand the areas where the vehicles can and can not drive. It has to be noticed that in real world scenarios these critical systems have no chance to rely exclusively on external position measurements. Considering that the necessary vehicle localization accuracy for a safe autonomous driving is estimated being on the order of 10 cm [2], the availability and reliability limitations coming from the even most sophisticated Global Navigation Satellite Systems (GNSS) cannot reliably guarantee lane-level localization accuracies, *i.e.*, a sufficiently precise localization to avoid catastrophic failures. On the other hand, we have to consider the undeniable potential arising from other information sources such as the cartographic maps, whose feature set can be leveraged as a priori knowledge in a large amount of feature detector. It is worthwhile to consider that the number of features provided by these services is nowadays sufficient for the vehicle localization process, specifically in dense urban areas where city-wide projects like Open LiDAR[2] or the upcoming high-definition maps, especially designed for autonomous cars, are candidates to allow vehicle localization systems to outclass the accuracies provided with GNSS signals only. Although some ADAS systems are today already commercial products in the automotive industry, the localization problem still needs a definition of a standard set of perception capabilities and a method to handle the observed features. Towards this goal, in this thesis we present a technique to synergically integrate a heterogeneous sensing pipeline within a single framework of perception.

1.2 Problem Statement

Given a vehicle in one of the proposed driving scenarios, our purpose is to provide a reliable localization estimate without experiencing typical GNSS accuracy degradations. We propose to exploit standard cartographic maps, to integrate their extended knowledge base, initially conceived for human users' applications. In details, we jointly tackle both highway and residential contexts, as they place similar challenges yet

[2] https://rapidlasso.com/2017/01/03/first-open-lidar-in-germany/

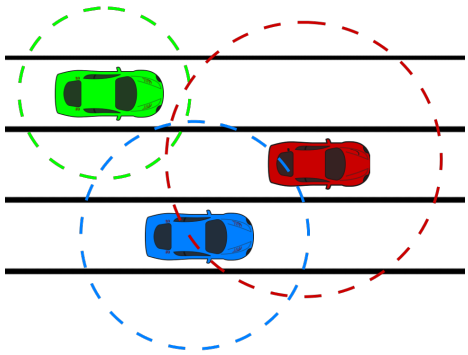


Figure 1.1: Typical localization accuracy resulting from GNSS systems. In urban areas, their signals are often weak or greatly corrupted by multi-paths. Consequently, vehicle localization within a lane is not possible. Please notice that the accuracies are not constant and vary also from receiver to receiver, preventing a safe cooperative driving using GNSS-based systems only.

simultaneously requiring different approaches with regard to the vehicle pose estimation issue. For this purpose, our system exploits a stereo video stream, as camera applications to the mass market is not hampered anymore by high equipment price. From a technical perspective, we tackle the vehicle localization problem proposing an on-line framework designed to handle the localization uncertainties that can arise in both urban and highway scenarios. These environments allow us to set a first interesting consideration, which is strictly connected with each driving field: which features could be exploited to enhance the localization estimate? Are they sufficient? In this thesis, we try to solve this issue by introducing into our framework four different sensing pipelines, aimed at detecting different elements of the vehicle's surroundings. The detectors are specifically designed to leverage specific characteristics of the scene, in such a way as to allow the localization framework to exploit the detections in a global perception system.

1.3 Challenges

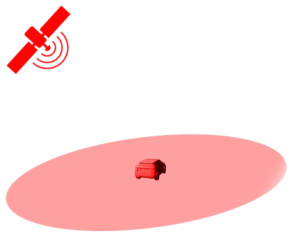
Instead of relying on a monolithic framework capable of a total scene understanding, we aim at creating a probabilistic scheme that enables a productive integration of information generated by any kind and any number of sensors. In the context of autonomous vehicles, achieving a reliable and adequate vehicle localization presents multiple challenges

which are, in most cases, related to the environment perception. On the one hand, the availability and reliability limitations coming from GNSS systems, which can give us no more than an in-road estimate, as depicted in Figure 1.1, require further detection pipelines in order to create a description of the surrounding scene. The main issue in detecting relevant features in common driving scenarios using computer vision algorithms is due to the variability of the features and the different environmental conditions. This is even more challenging given the huge amount of clutter presents in complex areas such as cities.

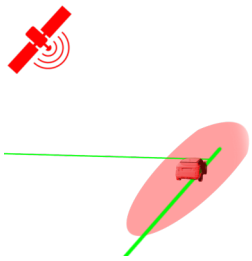
While a large number of approaches concerning the detection of singular scene elements, such as road markings and lanes [3, 4, 5, 6, 7, 8], cartographic approaches [9,10,11,12,13], buildings [14,15,16,17,18,19] and intersections [20, 21, 22, 23, 24] were proposed over the recent past, these features alone are usually only robust in specific scenarios, *e.g.*, urban canyons or highways. Thus, they cannot guarantee the designed localization accuracy on a sustained basis, *i.e.*, during a journey through different driving contexts as required in SAE Full Automation level. Consider, for instance, the situation depicted in Figure 1.2. A car traveling with only the GNSS signal has a typical localization uncertainty on the order of 10-20 meters [25], which is definitely not enough for lane-level positioning. A first attempt to reduce this rough estimate is to leverage the existing cartographic maps, allowing the system to place the vehicle, at least, on the road. However, the main challenge arising from this approach is related to the misalignments between the cartography and the real road lanes; moreover, no further information is here provided with respect to the longitudinal axis. One possible solution could be as follows. If the system was able to detect the road surface, *i.e.*, without a supplementary road marking analysis, we would be able to identify our distance from the center of an intersection. Adding the detections of the surrounding buildings would finally result in a lane level localization estimate, contributed by all of the aforementioned features.

1.4 Applications

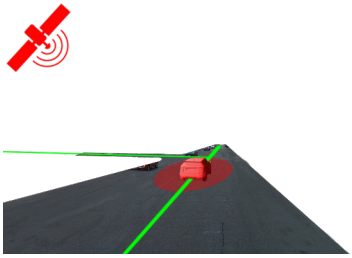
The purposes of the proposed system are not limited to the vehicle positioning task. Localization is one of the fundamental requirements for advanced applications like the strategic aspects of the driving task, *e.g.*, planning and navigation an automated driving system. Having a proper



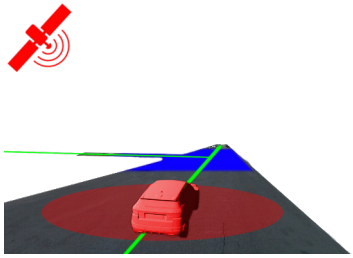
(a) Usual vehicle localization estimates, given the GNSS signal only, have uncertainties on the order of 10-20 meters



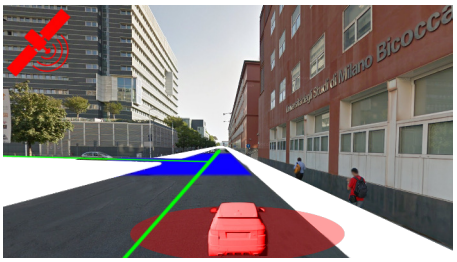
(b) Including a road graph in the localization process can help the system to reduce the uncertainty, achieving in-road level accuracies. Please note that the localization can be biased by misalignments



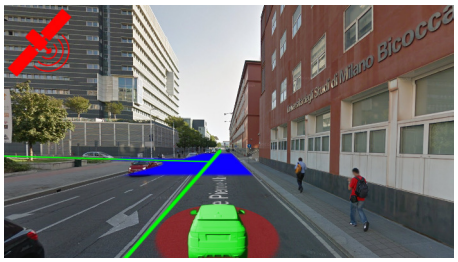
(c) If the system is able to detect the road surface and its boundaries, it will be able to center within it, detecting and correcting the misalignments bias. No improvements over the longitudinal axis are still achieved



(d) Detecting an intersection area like in this image, would help the system to reduce the longitudinal uncertainties. Moreover it would allow the system to discriminate between contrasting hypothesized estimates.



(e) In urban areas, adding the buildings' detection would result in a lateral improvement. Longitudinal improvements may also be achieved in proximity of intersection areas, exploiting buildings' on the opposite side



(f) Additional detections allow the system to compare the estimates with respect to each other, enriching the scene description and providing a scene understanding framework

Figure 1.2: The figure depicts the proposed scheme architecture used to tackle the localization problem. It consists in a probabilistic scheme that enables a productive integration of information generated by specific sensing pipelines.

localization within a component-wise system would allow us to exploit the detection set in a proactive manner. As an instance, a road marking detector may benefit from the results of a façades detection pipeline, by limiting the area to take into consideration. In reference to mapping services like OpenStreetMap, having a prior estimate over the number of current lanes of a hypothesized position may allow the system to adapt to specific illumination situation by automatically tuning its parameters and thus to increase its reliability and the achievable vehicle driving modes, *i.e.*, the type of driving scenarios. Finally, the results of the single detectors could be also used to update, validate or even integrate the features within the mapping services or, at least, to trigger automatic alerts to a mapping maintainer or map-managing authority.

1.5 Contributions

With respect to the aforementioned issues and challenges, the contributions of this thesis can be summarized as follows:

- We have introduced a novel framework for urban road layout estimation. It allows us to exploit a broad range of information sources, by translating them onto a common probabilistic basis. We demonstrate the flexibility of the proposed framework by exploiting different detection pipelines, dealing with both urban and highway scenarios.
- As opposed to other state-of-the-art works, our detection pipelines are on-line procedures. It follows that we do not need an evaluation of a whole sequence before including the detections into the evaluation framework.
- To leverage existing knowledge in the context of driving scenarios is essential. For this reason, every detection pipeline has been tightly coupled with the OpenStreetMap service, allowing the system to integrate priors into the evaluation.
- In order to enhance the ego-vehicle localization, we have proposed the following detection pipelines:
  - First, the effectiveness of the approach has been demonstrated by integrating the cartographic map provided by the OpenStreetMap service and limiting the vehicle position to stay on

the roads. We achieved excellent localization performances with respect to state-of-the-art and comparable algorithms.

- We have integrated a pure-geometric pipeline able to take advantage of the building façades detection. Differently from other state-of-the-art systems that hinge on image analysis, ours does not require a training phase. This component allows the framework to reduce both lateral and longitudinal localization uncertainties in urban areas, usually afflicted by GPS lack of precision due to so-called "urban canyons".
- We have proposed an intersection detection pipeline able to discriminate the road interconnection model leveraging both images and cartography. Unlike state-of-the-art systems, our on-line system does not require the whole sequence of images up to when the vehicle is inside the intersection before starting the computation.
- Towards obtaining an accurate lane-level global localization in highway scenarios, we have introduced a component able to leverage a line detection algorithm and the OpenStreetMap road features. Combining these information sources, the system is able to reduce localization uncertainties in highway scenarios allowing the system to cope with treacherous situations arising from inaccurate standard GPS measurements.

1.6 Thesis Outline

The work presented in this thesis is organized as follows. In Chapter 2, after a brief introduction to the history of autonomous driving cars, we provide a survey of the underlying techniques for vehicle localization, which are derived from both the robotics and the computer vision research fields. Chapter 3 introduces the proposed road layout estimation framework and the first three components that have been exploited to enhance the vehicle localization in both urban and highway scenarios. In Chapter 3 we propose a novel on-line method for intersection detection, while in Chapter 5 the presented algorithms are evaluated and critically discussed. Finally, in Chapter 6 we present our conclusions along with the future research perspectives that this work opened.

Figure 1.3: Localization in a residential scenario. The localization framework is currently tracking 80 vehicle position estimates (depicted with the yellow cars), leveraging both road graph (shown with red lines) and the building outlines data (green-cyan) from the OpenStreetMap mapping service. The blue area overlayed with the building's outlines represents our façades detections. In the bottom right, the components of the intersection detection pipeline. Finally, the fuchsia and green track represent respectively our best estimate with respect to the DGPS ground truth.

Chapter 2

Related Work

This chapter provides an overview of the existing state-of-the-art approaches for perception in the autonomous-driving scenario and, on the basis of the existing literature and current challenges, we focus on the main contributions of the proposed approach.

2.1 Autonomous Driving

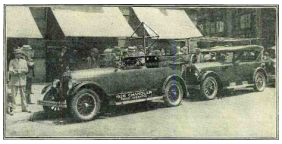
The human ambition of having automated driving cars dates back to the beginning of the past century, when in 1925 and 1926 the first driverless vehicles were presented in New York and Milwaukee. Despite the *Linrrican Wonder* and the *Phantom Auto* were controlled by radio signals sent by other following vehicles and so were not autonomous, they were a practical verification of the available technology at that time. A decade later, during the 1939 World's Fair, General Motors (GM) presented its vision of the next future at the Futurama exhibition Figure 2.1b. Illustrating their opinion about the *New Horizons*, GM presented a new motorway concept where distances between cars were maintained by automated radio control systems, and the curbs were allowing the drivers in keeping the car within the proper lane, gaining safety concurrently with high vehicle speed. Even though these first vehicles were mainly bounded to experimental activities, the safety and optimization benefits arising from removing humans from driving tasks were already clear. In the next decades, research in vehicle's automation included electronic controlled highways and automated highway systems (AHS), with contributions from both the academic and the automotive industry. How-

ever, the technology behind this generation of automated vehicles was far from the current approaches, because of the absence of any vehicle autonomy.

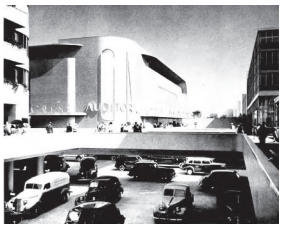
With the technological advancements of computer vision, machine perception and computation power, in the beginning of the 80's the team of the Bundeswehr University Munich lead by Professor Ernst Dickmanns started the research in autonomous driving. One of the first *modern* vehicles equipped with on-board sensors and computer systems, was a Mercedes Van called with the acronym VaMoRs (Versuchsfahrzeug für autonome Mobilität und Rechnersehen), which was presented in 1986 in Germany. It successfully performed a first autonomous driving experiment on streets without traffic, traveling at speeds of 96 Km per hour, limited only by engine speed over distances of 20 Km [26]. Approximately at the same time, the Navlab laboratory of the Carnegie Mellon University proposed its first road navigation platform, Navlab 1 [27], starting the research on intelligent mobile robots capable of operating in the real world outdoors [28, 29]. Since then, complex algorithms constantly increased the autonomy level of autonomous research cars and, at the end of the EUREKA Prometheus project (1987-1995), the Dickmanns's VaMP vehicles showed the fully autonomous capability of lane change on a three-way highway with normal traffic.

Comparable results were also achieved by the ARGO project of Professor Alberto Broggi in the last years of the 90's [30]. His team's vehicle drove the *Millemiglia in Automatico* tour, consisting in driving approximately 2000 Km along the Italian highway network [31], most of times in full autonomous mode.

The aforementioned approaches were mainly focused on enhancing vehicle autonomy in highway scenarios. The US Defense Advanced Research Projects Agency (DARPA) promoted the Grand Challenges in 2004 [32] and 2005 to tackle more complex road conditions. These competitions raised the interest of the robotic and computer vision academic communities in applying their research to more complex and realistic environments. Although no competitor finished the first challenge, which took place in 2004, in the 2005 edition the Stanley vehicle of the Stanford University lead by Professor Sebastian Thrun, won the run achieving a 212 Km autonomous driving [33]; other four vehicles completed the challenge. The competitions were held in off-road desert terrains and required developments in both terrain perception, real-time collision avoidance and vehicle control [33]. Two years later, in 2007, during the



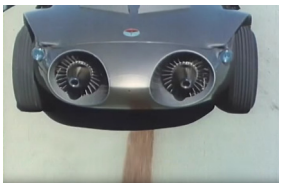
(a) The Linrrican Wonder, 1925



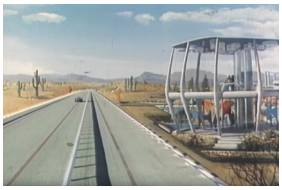
(b) The Futurama Exhibit, World's Fair 1939

(c) New Horizons, General Motors, World's Fair 1940

(d) Firebird I Prototype



(e) Firebird II, General Motors's Motorama exhibit in 1956



(f) Tower Control at Motorama exhibit in 1956

(g) Prof. Dickmanns's VaMoRs Project, 1986

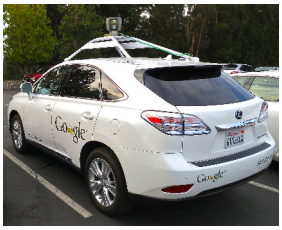
(h) Prof. Dickmanns's VaMP Project, 1994

(i) Prof. Broggi's ARGO Project, 1996

(j) Stanley, the Grand Challenge winner, 2005

(k) The Sandstorm during the 2005 Grand Challenge

(l) Boss, the Urban Challenge winner, 2007



(m) The Google Self driving vehicle

(n) Tesla Autopilot

(o) The Uber Self driving vehicle

Figure 2.1: Progress in autonomous driving research from 1925 to 2016.

third Grand Challenge [34], the agency increased the difficulty level of the challenge and held a new driverless car competition in a simulated urban scenario with emulated traffic. The race, known as Urban Challenge, required a new set of algorithms specifically tuned for urban-like environments [33, 35].

In all the three DARPAs events, the focus was on controlling the vehicle in a pre-defined environment, *i.e.*, given a map and a concise plan, the vehicles needed to safely drive to the goal understanding the nearby environment using its on-board sensors.

In 2009, Google started its self-driving car project [36], bringing together the most experienced engineers who had been working with autonomous cars including Sebastian Thrun, Chris Ursom, Mike Montemerlo and Anthony Levandowski from Stanford and Carnegie Mellon. The company significantly pushed the boundaries of the researches done for the past DARPA competitions, relying upon laser measurements matched against pre-recorded maps, which were collected using manually driven vehicles [37].

In the past years, the efforts concerning the development of intelligent transportation systems (ITS) such as intelligent vehicles increased significantly. Nowadays, a large number of academic laboratories as well as established automotive companies, along with new innovative companies, *e.g.*, nuTonomy, Tesla Motors, Uber Technologies etc., have started their driverless car projects, and while Advanced Driving Assistance Systems (ADAS) like auto-parking, lane-keeping or collision avoidance systems are already available on the market as options, they get mandatory in the next decade [38] following a path similar to ABS/ESP systems, which are mandatory nowadays. Moreover, systems aimed at improving security including Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication infrastructure have been proposed and are likely to be the next forthcoming innovations.

2.2 Localization

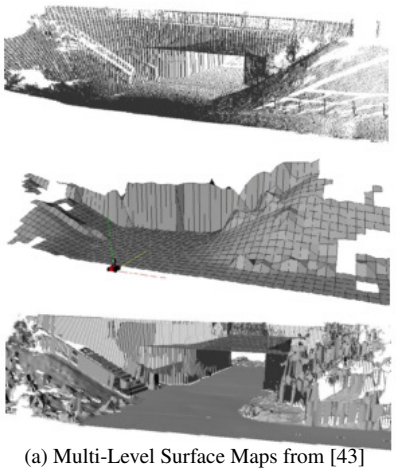
The localization problem is a well-known issue in autonomous mobile robotics and it has been tackled by using approaches arisen from different research fields, not robotics-related only. From a technical perspective, the mobile robot localization problem *"is the problem of determining the pose of a robot relative to a given map of the environment"* [39].

It follows that the first key aspect is related to the existence and the availability of the map in which the robot has to localize itself. A second but of equal importance aspect is related to the environment perception. This aspect includes the issues related to the specific sensor typology used to perceive the environment. Starting from the latter concept and assuming that we do not need to create a new map from scratch, a remarkable distinction between localization algorithms should be made, *i.e.*, whether the robot needs to perform a localization process in an indoor or outdoor scenario. Since the environments have a strong impact on the map typology, the next sections present an overview of the existing mapping approaches regarding the autonomous vehicle's context.

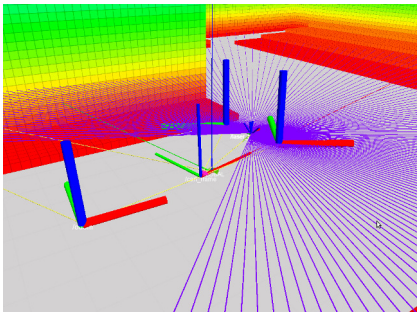
2.3 Grid-Based Maps

The first and widespread map typology we consider is the occupancy grid map. This kind of map stores a binary value inside each of its elements, representing exclusively whether the space area element is occupied or not. Despite this trivial representation, these maps are commonly used in indoor mobile robotics because such environments allow us to use 2D-3DoF maps and thus to simplify the feasible state-space of the robot. Usually the state-of-the-art approaches [40, 41, 42], mainly designed for indoor robotics, exploit this conjecture by mapping the environment to two-dimensional grid maps, which are then used along with laser or sonar range finders to perform the localization. One of the advantages of this representation stems from the seamless integration of this kind of sensor measurements.

Extensions of 2D grid mapping for robots working in non-flat terrains were also proposed. In [44, 45] two-dimensional grid maps were enhanced to store the height information of each space element. However, these maps, known as Digital Elevation Maps (DEM), do not allow us to accurately represent vertical structures where multiple levels of height are semantically relevant. These structures are quite common in driving scenarios, *e.g.*, bridges, tunnels, and multilevel parking lots. The limitations of these approaches arise in the map generation phase, where the stored height value is calculated averaging all the measurements of a specific mapped area. Moreover, such average makes the localization process harder, since the stored elevation may be significantly different from the sensor measurements gathered during the localization phase.



(a) Multi-Level Surface Maps from [43]



(b) Full 3D voxel maps

Figure 2.2: The pictured depicts mapping examples using Digital Elevation Maps and Multilevel Surface Maps [43] in 2.2a and full 3D voxel maps in 2.2b

To deal with the aforementioned environments the authors in [43] proposed a new further enhancement by means of a representation known as Multilevel Surface Maps (MLS-maps), depicted in Figure 2.2a. This time, each element of the grid consists of a discrete list of surfaces associated with it. Such map representation was widely used in [46], and permitted the authors to avoid complex representations of a full three-dimensional map. In [47] the authors applied MLS-maps to solve the localization problem inside a multilevel parking structure, introducing a 5DoF motion model that limits the z-value of the vehicle to the height indicated within the map.

In spite of this restriction, a full 3D-6DoF motion model not relying on the structure of the ground surface was introduced in [48] and depicted in Figure 2.2b. It is worth noting that the modeled motion does not consider the interactions between the errors acting on its basic components, introducing uncertainties on each single component of the movement according to a velocity model. It has to be observed that the independence between the single components of the pose is also commonly assumed in other works [49].

Finally, in [50] a full three-dimensional occupancy map model was introduced to provide a volumetric representation of space, which is important for a variety of robotic applications including flying robots and robots that are equipped with manipulators. This close our short review with respect to grid-based maps.

Remarkable work using grid-based mapping for localization was in-

troduced in [49, 51], where two-dimensional maps of laser scanners reflectivity values were stored, instead of occupancies values.

Although grid-based maps allow us to directly map the occupancies of the scene objects into a location-based representation, the physical scene-to-map associations put strict interpretation constraints. It can be concluded that the lack of a higher semantic interpretation level expresses the main underlying flaw of this map representation. As an example, these maps can not deal with high-level concepts like road lanes or street lines, except through an explicit position mapping, which it is hard to achieve in real-world dynamic scenarios. Despite the limitations, their real application strictly depends on the designated application. Even though benefits may arise in enclosed areas, *e.g.*, parking lots, these maps can not deal with high-level concepts like road lanes or street lines, except through an explicit position mapping, which is hard to achieve in real-world dynamic scenarios.

2.4 Feature Based Mapping

*Features map*s represent a second class of maps, where the stored elements contain high-level properties of a feature as well as geometrical terms. This map category allows us to combine qualitative and geometric clues, ensuring a suitable level of expressiveness for complex environment representations, while keeping them open to further enhancements. Even though the spectrum of the features included in these kinds of maps varies according to the requirements, approaches considering traffic scenarios present a definite trend that include elements to cope with the traffic regulations. Lines, lanes, roads and interconnections between them are just few of the obvious concepts which can be found in common urban maps. In this thesis, we perform vehicle localization by means of the interpretation of the environment surrounding the vehicle, which is then matched against an enriched cartographic map, retrieved from on-line mapping .

2.5 Topological Mapping

The maps handled by well-established cartographic services are an important piece of information that can be exploited in the automotive-related localization context. Even though no standard concerning fu-

ture maps for autonomous driving cars is yet defined, these maps, initially designed for human use, represent an incredible source of information that is nowadays also available in terms of on-line map services. From a technical perspective, efforts should be dedicated on leveraging the abundance of existing cartography as well as the forthcoming high-definition maps such as [52], specifically designed for autonomous driving cars and next generation *map-based* ADAS systems [53]. Following this motivation, in the last years authors have proposed a combination of information coming from on-line mapping services like Google Maps, HERE Maps, the collaborative OpenStreetMap or other Geographic Information System (GIS) projects, to better exploit the massive number of potential object classes and structures contained in the maps. These services have been used as a priori knowledge of the 3D scene structure [19, 54, 55, 56], usually referred to as *Scene Layout* [57, 58]. Unfortunately, even though we can expect good reliability from the upcoming commercial maps, the lack of strict guidelines and data validation rules often reduce the quality if the maps. To deal with missing or noisy information, the authors in [59] propose to enhance the annotation of existing roads and in [60] they also detect new roads, not yet mapped in OpenStreetMap. Other works, aimed at enriching the map with additional high-level concepts like *lanelets* have been presented in the literature, see, *e.g.*, [61]. Here the authors introduce a novel specification for autonomous driving maps, which allows them to also include traffic regulation rules. These elements are known as tactical information, grounding on the OpenStreetMap service.

Apart from the works aimed at improving existing maps, valuable contributions to the field came from the computer vision community, where the semantic image segmentation of road scenarios has received considerable attention [62, 63, 64, 65, 66, 67, 68]. These works combine different information about the context by means of probabilistic graphical models like Markov Random Fields or Conditional Random Fields [69]. Although these approaches lead only to an image partitioning in disjoint and classified areas, the resulting interpretations can be further exploited. As an example, the outcomes can be integrated, as new features, in existing maps or even used to perform a better localization by means of a semantic matching, rather a than purely geometric matching like usually happens.

2.6 3D Scene Understanding

For a better interaction with the surroundings, autonomous systems need a semantically high level representation of the environment [20, 70, 71]. The introduction of semantics for recognized objects allow the research community to go one step further with respect to the image based segmentation. As an example, by including high-level clues about the real world we may estimate the navigability boundaries of the surrounding space or even a localization position by reproducing the human ability of inferring the global structures and situations of an observed scene.

Many works have been proposed to solve the scene understanding problem: from the simpler scene indoor understanding process by means of single images, to the analysis of multiple images or sequences [64, 65, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81], the problem has been tackled both from the machine learning and the probabilistic perception side.

To recover the layout of indoor scenes by understanding its structure, geometric clues such as vanishing points were used, *e.g.*, in [75, 82]. These algorithms usually rely on the Manhattan World Assumption, *i.e.*, the orthogonality constraint of scene planes. To deal with cluttered scenarios, the authors in [73] modeled the 3D area by means of a *cuboids* representation, and generated layout hypotheses by sampling geometric directions estimated by using vanishing points. In [76] Lee *et al.*enhanced their previous works introducing volumetric reasoning, which allows the authors to model the 3D interaction between the objects and the spatial layout. Authors in [79, 83] achieved state-of-the-art results proposing a decomposition of the high order potentials used in [73, 76, 84] by means of the new integral geometry concept. Note though that the aforementioned approaches tackle the problem of understanding the indoor scene relying on single image analysis rather than on video sequences. Moreover, with the exception of the works by Urtasun *et al.* [83], most of other works do not take into consideration the time performances. In [85, 86] the authors propose a Bayesian filtering framework for dynamic visual understanding of the local environment. The idea behind their framework, which is also shared by the work in [70, 80], and also in the work proposed in this thesis, is to drop the hard Manhattan assumptions, the extensive learning phase, and the graphical model approaches, in favor of a filtering approach, able to efficiently integrate new evidence in order to generate, evaluate, and refine 3D layout hypotheses.

Even though the approaches mentioned so far aim to reconstruct indoor environments, a large amount of effort in the literature have been focused on the reconstruction of outdoor scenarios. Similarly to what has been done for the indoor context, the methods can be separated in two groups. On the one hand, some approaches, aimed at estimating the layout from single images, often rely on the Manhattan assumption [82, 87] or horizon and zenith lines [88]. In order to generate 3D models combining surface reconstruction and object recognition, the authors in [89] proposed a semantic scene analysis system, which introduced high-level constraints in an explicit knowledge base implemented as a semantic net. Pursuing the need of an effective scene understanding system, which can reason about the interaction and relationships between objects, the authors in [58, 90] proposed a framework to recover the surface layout of a scene by means of multiple image clues, spatial support and decision trees classifiers, extending their work in [91]. Following the idea of exploiting relationships between objects along with physical, mechanical, and geometric constraints, the authors in [92] extended the pioneering blocks-world concepts proposed by Roberts [93], by introducing a search strategy to determine blocks configurations consistent with the input image. From a technical perspective, the algorithm works by placing hypotheses over the valid configurations and then scoring them by weighting mechanical and geometric properties of each candidate configuration. Similar contributions were also achieved in [94] by Saverese *et al.*Although the original algorithm does not exploit video sequences nor explicitly models the features typically seen in driving scenarios, in [95] and [68] the authors introduced respectively visual semantic for street level imagery and reconstruction of 3D semantic models from urban environments. An alternative representation of the 3D space is also defined in [96,97,98], where a *stixel-world* is introduced, *i.e.*, a representation encoding the obstacles and free space of the current traffic scene. In [99] the authors propose a semantic segmentation method that does not rely on appearance-based descriptors, although it leverages 3D point clouds derived from a structure from motion (SfM) approach, onboard a moving vehicle. In [100] the authors propose leveraging geographical priors to achieve a richer outdoor scene understanding by jointly reasoning on 3D object detections, vehicle pose estimation, and semantic segmentation. The authors in [21, 101] address the problem of segmenting urban street scenes into semantically meaningful classes, *e.g.*, road surface, buildings, road markings and cars, leveraging temporal integra-

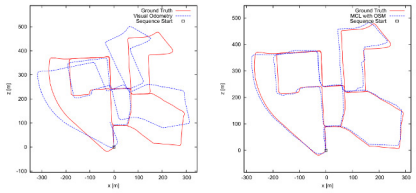
tion between consecutive video frames and focusing on the upcoming road section and the presence of common objects.

Keeping in mind the integration of dynamic objects and typical traffic scenarios, the authors in [102, 103] developed a reliable system capable to infer the geometric and topological properties of intersections, as well as the activities occurring nearby them, by means of a generative model and reversible jump Markov Chain Monte Carlo schemes that reason about static and dynamic objects. Furthermore, the scene understanding problem and high-level semantic segmentation have also been tackled in [104] and [21], where traffic images are analyzed in order to infer the road topology and the existence of an a-priori defined set of objects, as well as traffic patterns.

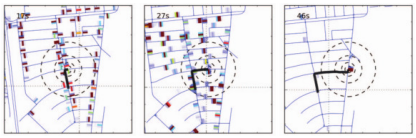
Despite the fact that the algorithms mentioned above were designed to reconstruct the best layout configuration of the scene, they do not directly exploit the detected scene objects for any specific purpose relative to the autonomous driving domain. In this thesis, we aim to take advantage of the latest advancements of scene understanding algorithms (*e.g.*, road segmentation and 3D interpretation of crossing areas and buildings) to achieve a good localization accuracy with respect to the geo-localized entities available in both urban and highway areas.

2.7 From Scene Understanding to Urban Localization

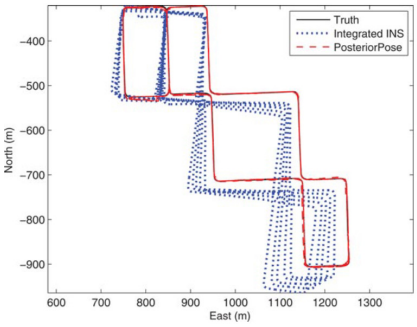
In the past years, the efforts for the development of intelligent vehicles increased significantly. Systems aimed at improving security, including Advanced Driver Assistance Systems (ADAS), Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication, were introduced. Many of the ADAS systems rely on object detection and tracking or on scene understanding techniques. They all require an accurate localization within a known map [9], in addition to a rich description of the surrounding scene in terms of pedestrians, cars, traffic signs, road lanes, etc. To deal with the localization problem, researchers from both computer vision and robotics developed different approaches. A common goal is to achieve a good localization accuracy, despite the limitations of Global Navigation Satellite Systems (GNSS), which is frequently unreliable, degraded or full absence of signal because, *e.g.*, of urban canyoning. Some authors already proposed to leverage information from mapping services like Google Maps, HERE Maps or the col-



(a) OpenStreetSLAM, Floros et al. 2013 [11]



(b) Global city-wide localization by means of visual odometry and road maps, Brubaker et al. 2013 [105]



(c) Map-aided localization, Miller et al. 2011 [106]

(d) Localization using visual odometry and digital maps, Parra Alonso et al. 2012 [9]

Figure 2.3: Progress in vehicle localization, leveraging topological road graphs such as OpenStreetMap.

laborative OpenStreetMap project. So far, the most exploited information is the road graph, which gives both topological and metric (known as topometric [12]) clues to localization. The road graph is used to narrow the localization uncertainties, by zeroing the estimated distance between the vehicle and the nearest road segment by means of a *lock-on-road* procedure [9, 107], as well as ad-hoc schemes in intersection areas. In [108] the authors leveraged the metric properties of maps and a visual odometry input, exploiting a chamfer matching technique to align the path traveled by the car to the road network. Similarly, the authors in [109] extended their previous works [110], and replaced the visual odometry input with the readings from the Anti-block Braking System of a common vehicle; moreover, they applied a tightly-coupled fusion strategy to integrate the GNSS measures. Similar results were achieved in [54, 111, 112]. Lu et al. [113] perform localization by coupling vision-based detection of lane markings with open source map databases. Recently, Larnaout et al. [16] proposed to use the building models provided in OpenStreetMap as geo-referenced antennas in order to correct the GPS inaccuracies. Additional contributions, *e.g.*, [114], analyze the behavior of the tracked vehicles in order to understand the intersection

structure. Beside the road graph, the authors in [115] propose a method capable to perform localization leveraging on-board images captured by a common mobile phone and exploiting Google StreetView imagery. Despite its coarse accuracy, *i.e.*, they do not achieve accurate in-lane localization, maybe one of the most impressive result on localization is presented in [105], where the visual odometry is exploited to localize a vehicle without any GPS prior in more than 2000 km of driving roads in just a few seconds. A technologically different approach not using imagery but laser sensors, was recently presented in [19]. Here the OpenStreetMap data is again used as a prior information within a supervised classification approach, in order to classify road and non-road readings.

A complementary category of approaches tackle the problem of understanding the position of the camera in terms of global localization, given single images. These methodologies, known as location or place recognition [116, 117] and binary codes extensions like [118], aim at recognizing locations from their appearance rather that exploiting consecutive images taken from moving platforms.

Even though the object perception field spans heterogeneous targets, this thesis, starting from road appearance clues, to the presence of other vehicles, leverages the most distinguishing features to infer the structure of urban and highway settings, *i.e.*, their main structure. The detected features are used in order to reduce the localization errors, by exploiting their pose in the common reference frame in OpenStreetMap.

2.8 Sensing the Environment

Autonomous vehicles hinge on advanced algorithms for object detection and tracking, for self-localization, and vehicle control. Although each of these components is essential to safely plan the vehicle actions, all such algorithms concurrently support the main challenge for autonomous vehicles, *i.e.*, understanding of the surrounding environment. Consequently, the perception and the interpretation of the objects and entities within the scene is crucial, since a proper interpretation could prevent the vehicle from running into potentially treacherous situations.

Even though multiple domains may benefit from accurate 3D urban models, such as the entertainment or digital mapping domain, the challenge of building them automatically has been tackled by researchers from both robotics and computer vision fields, and it is strongly re-

lated to the scene understanding process and its geometric reconstruction [15, 68]. From a technical perspective, the approaches to scene understanding can by distinguished in two main categories, depending on the sensing devices that are used to perceive the scene.

The first category rely on the highly accurate laser scanners [119, 120, 121], and external data sources like cartographic maps, combined with GNSS data [9, 16, 19, 108]. An interesting approach based on laser scanner for exploiting the building outlines in the well-known SLAM problem, is presented in [122]. Although the algorithms relying on laser scanners may benefit from the great accuracy of the measures, their applications to the automotive market are still hampered by the equipment price, even though a lot of innovation is happening in this field. Companies like Ford and Baidu have recently signed an agreement to help the development of cost effective laser scanners sensors[1]. On the other hand, the problem can be tackled by leveraging visual clues, in order to assemble a good feature set, which is then used to train different classifiers. Although in the recent years heterogeneous techniques, including Conditional Random Fields (CRF), Decision Trees, Description Logics or Deep Neural Networks [101, 123, 124, 125, 126, 127], have been used to tackle the scene understanding problem, it still remains a remarkable challenge for the research community, as proved in recent works such as [13, 16, 19, 128].

In both cases, the researchers have focused on approaches that allow the detection of both static and dynamic objects of the reconstructed environment.

Even though, conceptually, object perception spans heterogeneous targets (from road appearance to other vehicles), we focus on the most distinguishing features of two typical driving scenarios, *i.e.*, highways and urban areas.

2.8.1 Buildings

In the context of urban driving scenarios, the understanding of the scene near a vehicle is a complex task, due to the nature of the environment, *i.e.*, presence of a large degree of clutter. Dense traffic circumstances as well as the city infrastructure elements can further complicate the scene understanding process. In order to effectively tackle this task, a

[1] http://spectrum.ieee.org/cars-that-think/transportation/sensors/ford-and-baidu-invest-150-million-in-velodyne-for-affordable-automotive-lidar

common approach is to leverage methods for scene segmentation and interpretation.

Although the research has so far mostly been focused on approaches that leverage intrinsic road features, including ground plane and surface orientation analysis [58], together with road markings, lane, and curbs identification [119,129,130], an interesting, yet challenging, opportunity in the context of urban areas arises from the exploitation of the intrinsically static elements, *e.g.*, the buildings. Regarding this specific domain, the existing attempts vary according to the different applications.

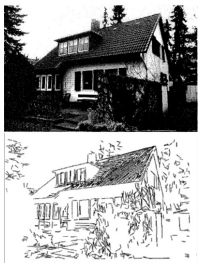
A comprehensive survey of the possible approaches is available in the excellent work of Musialski et.al. [15]. To deal with this complex task, advanced techniques like Conditional Random Fields (CRFs) [131] or decision trees based algorithms are often used to learn appearance-based models of geometric classes [58].

In [132], the authors propose to extract the 3D properties of the buildings by inferring the façade planes from single images and leveraging the properties of straight lines and vanishing points. In [14, 128], the authors propose a probabilistic discriminative model called BMA, where descriptive features are attached to a set of aggregated regions. By means of a twofold consideration, the authors try to answer the following question: *Is this pixel part of a building façade, and if so, which one?* Firstly, the BMA approach is used to label the pixels in the image. Then a set of candidate planes is generated by sampling the image and performing Principal Component Analysis (PCA) to approximate the local surface normal at the sampled points. Finally, both information are incorporated by means of a Markov Random Field, allowing the authors to answer the aforementioned question.

Moreover, following the semantic scene analysis system presented in [89], the authors proposed to model 3D Buildings using High-Level Knowledge [133].

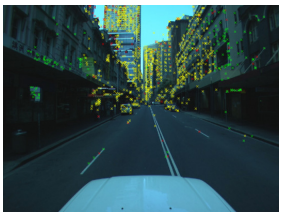
Although many authors have put significant efforts in detecting buildings even from a single image, it is clear that inference gathered from an in-vehicle stereo image stream allows us to place object hypotheses, *i.e.*, 3D building models, in a way that can be corroborated by temporal integration [71].

However, even though the aforementioned approaches yield good results, our goals are slightly different and closer to the robotics domain. One of the most important problems that may benefit from a high-level interpretation of the scene is the self-localization problem. It repre-



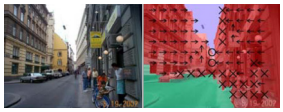
(a) 3D Modeling with constraints, Grau 1997

(b) Planar Surfaces in Outdoor Urban Environments, David 2008

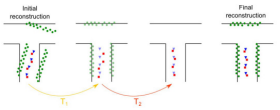


(c) Vision-based self-localization with Buildings, Himstedt et al. 2012

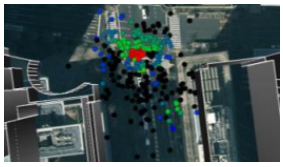
(d) Is the pixel a building-pixel? Corso 2008



(e) Surface Layout from an Image, Hoiem et al. 2007

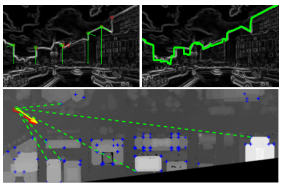


(f) Improving VSLAM 3D Building Models, Larnaout et al. 2014



(g) 3D Building Map for Lane-Level Localization, Gu 2015

(h) 3D Point Clouds to 2D Maps, Ni et al. 2013



(i) Geometric Urban Geo-localization, Bansal et al. 2014

Figure 2.4: Progress in building detection, using different approaches.

sents a critical task in every autonomous system [40] and this holds true even more in the context of autonomous transportation systems, where slightly erroneous positions and orientations of the vehicles could have a strong impact on the whole system safety.

Many image-based approaches, such as the one proposed in [116], are closely related to *place recognition* rather than to metric localization. Nevertheless, in the last years, a number of approaches, leveraging different cues like cartographic [134, 135, 136, 137] and building maps [9, 12, 18, 108] have been proposed to compensate the lack of metric precision and reliability of the GNSS receivers. Despite the achieved outcomes, an accurate lane-level localization is still a challenging task to achieve.

In [17] the authors tackle the problem of geo-locating images taken from moving vehicles using only basic geometric features of the buildings (*e.g.*, roof-line edges). In [13] 3D laser scan readings are pro-

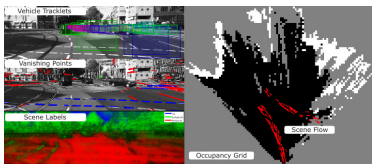
cessed with a Virtual Scan method [138] before being matched against the OpenStreetMap landmarks. A similar approach based on laser readings was developed in [139] where the authors do not rely on a mapping service, but they rather leverage the aerial imagery to extract the building edges, which are then matched against the upper edge of the buildings detected with a laser scanner. In [16] the authors propose a "Differential-GPS" based on building models. The algorithm exploits a GPS correction module in order to improve a Visual Simultaneous Localization and Mapping (VSLAM) procedure.

In this thesis, we propose to leverage the façade's geometry of the buildings in order to enhance the localization of a urban vehicle. In contrast to approaches aimed at explicitly detect façades using image processing techniques (*e.g.*, using symmetry or texture analysis or pattern recognition), our approach relies on stereo vision from image pairs only.

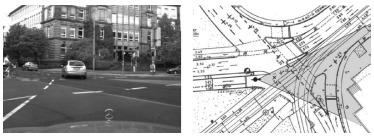
2.8.2 Road Intersections

The first studies about the detection and modeling of the road intersections date back to the end of the '80s, with the works of Kushner and Puri [140]. The authors proposed to detect the intersection geometry by matching the detected image road boundaries with a predefined intersection model derived from an *a priori* map database. A second approach using laser range finders was also proposed in their work. Although approaches using high-end laser scanners like the Velodyne HDL-series benefit from accurate measurements, achieving state-of-the-art results in both intersection [141] and roundabout recognition [55], their application to the real automotive industry is still hindered by the cost. On the other hand, vision-based system were introduced, *e.g.*, in [23, 142, 143], where the authors exploit digital maps in order to generate road and intersection models that can be used by model-based tracking algorithms.

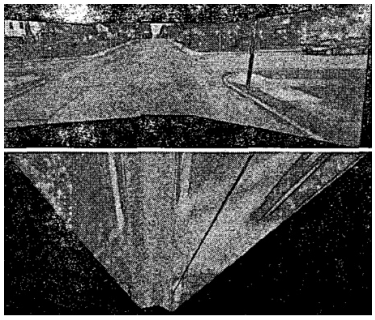
Recognizing intersection using on-board imagery only was defined as a hard problem, see, *e.g.*, [22,23,144]. Most algorithms rely on structured road detection, where features like borders or line detections [144] are exploited in order to determine a high-level classification of the geometry of the road ahead [24]. Although this may be an effective approach for vehicles traveling on ideal and loosely crowded roads, in the context of urban scenarios it simply fails because of the nature of the assumed environment, *i.e.*, the presence of a large degree of clutter. Beyond these initial attempts, authors in [22] proposed a method that hinge



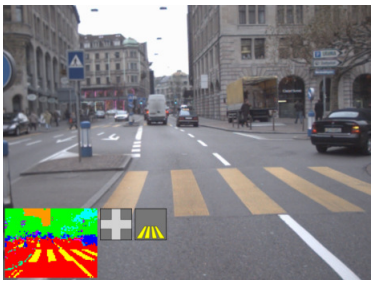
(a) Approach by Geiger, vision based



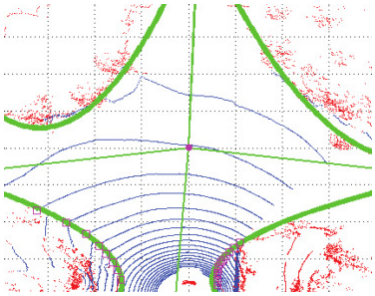
(b) Approach by Hummel et al., description logics



(c) Approach by Rasmussen et al., vision based



(d) Approach by Ess et al., vision based



(e) Approach by Zhu et al., laser based, 2012

(f) Approach by Zhang et al., laser based, 2015

Figure 2.5: State-of-the-art approaches for intersection detections. While laser based approaches leverages geometric clues, vision algorithms mainly rely on pixel appearance and inference schemes.

on road-surface detection, which is evaluated using color classification and road typology matching. They introduced an intersection model in the SCARF system that even works in case of degraded surfaces, *i.e.*, with missing lane markings or in difficult shadow conditions like in Figure 2.6. In [23], the authors added a geometric model of the lane structure so as to specify the lane width and the distance between the road junctions. Their model is based on the assertion that even a rough intersection model can significantly reduce the uncertainty over the longitudinal position, which is also a cornerstone of our work. Many of state-of-the-art algorithms tackle the road intersection problem as a by-product of a more complex scene understanding scheme. Moreover, even if the object classification results were promising, the outcome of the intersection identification highlighted the presence of several problems, in particular with the classification of the different types of junctions. Approaches like these in [57,58,101,145] are closer to our approach. In these works, the authors used both CRFs to detect street scenes and surface layouts, as well as temporal integration schemes to take advantage of the coherence between the road models computed in consecutive frames achieve stable temporal detections. Andreas Geiger's dissertation thesis [20] may be considered one of the most exhaustive works on intersection detection. The author extracts information about the intersections tackling the full layout interpretation in a probabilistic fashion, and exploiting video sequences from 5 to 30 seconds in length. The system leverages vehicle tracklets, vanishing points and semantic labels, road parameters as well as scene flow by means of a probabilistic graphical model (PGM) which allows the algorithm to cope with complex situations not discernible by single detectors. Notwithstanding the excellent results achieved, the approach shares with the aforementioned works the off-line strategy, *i.e.*, they all reason about the intersection geometry only after a first image sequence has been processed. Conversely, in this work we propose an on-line detector of road intersections that does not require the whole sequence of images, *i.e.*it works on-line, up to when the vehicle is inside the intersection. We then use the resulting output to disambiguate the treacherous localization circumstances arising in typical urban scenarios, by means of matching the detected crossing topology with the OpenStreetMap data.

2.8.3 Road Features Detection

One of the major challenges for high-level algorithms is related to the detection of the road surface. Achieving a good road surface detection is crucial since it is the basis for more complex algorithms, *e.g.*, navigation and vehicle control, but its detection is usually limited by the significant amount of clutter that is usually found along the roads. On the one hand, vanished road markings, unusual or specific weather conditions, or even light variations may complicate the road surface detection, as depicted in Figure 2.6. On the other hand, the visibility of the roadway may be frequently hampered by the presence of other vehicles, thus requiring other considerations to solve the problem. From a technical perspective, the approaches can be distinguished basing on the sensor used to perceive the road surface. Even though the approaches relying on laser scanners, which are mainly used to detect road limits and markings, are able to detect obstacles up to 100 meters with a 360° field of view, vision-based algorithms may outperform them in the perception of the road appearance. Laser scanners can be characterized by the number of concurrent acquiring layers.Despite single layer laser scanners are normally used to detect obstacles in the vehicle's surrounding, authors in [49, 146] showed that road features can be also detected and used in order to efficiently localize a vehicle. Rather than using the range measurements only, they accumulate multiple reflectivity readings over time, generating a *swathe* of measurements that are used to match against a prior survey. In these approaches the laser scanners sensors are mounted downwards in order to detect the ground plane, and a vehicle motion compensation is required to correctly generate the representation of the road surface from the data gathered from by the sensors. Moreover, the reflectivity information can also be used to retrieve information about the road markings. Besides single layer laser scanners, solutions with 4, 8, 16, 32 up to 64 layers can be found on the market. Also these high-performance sensors yield extremely accurate 3D reconstructions of the surrounding environment, which can be used for disparate purposes that span from the curb detection to the surface and road markings [141, 147, 148, 149, 150, 151, 152, 153].

On the other hand, algorithms relying on visual input can potentially leverage the huge amount of information contained in images. Regarding common driving scenarios, the different road typologies existing make the surface interpretation a hard task. Since the road appear-

(a) Sub-optimal road surface conditions

(b) Partially faded or hidden lines

(c) Faded lines

(d) Totally faded or non existent lines

(e) Bad illumination or weather conditions

(f) Shadows issues

Figure 2.6: Typical issues in road markings detection context.

ance may vary in many ways, as depicted in Figure 2.6, the different approaches usually tackle the issue as a segmentation problem, leveraging different color spaces [154], pixel intensities [155], texture [156] and other considerations including road markings identification [157] or even 3D layout configurations integrated over time [57]. In [158] the authors exploit a hierarchical method that use Gaussian Mixture Models (GMM), super-pixels and the GrowCut [159] algorithm for image segmentation. The results are then refined by means of a Conditional Random Fields [69] approach, which allows the authors to include road shape priors and thus a more robust detection. Super-pixel approaches were extensively used also in [160]. To better deal with the shadows and the variability generated by lighting conditions, the authors in [161] introduce a shadow-invariant feature space and a likelihood-based classifier, managing to achieve real-time performances in a per-frame basis. In order to cope with faded or not existing road markings and discontinuous or irregular curbs, the authors in [162] propose to exploit the combination of boundaries detections from the gray-level images with the color information of the captured image. The authors in [145], extending their previous work [163], propose to introduce road geometries (called road shape models) for road detection, exploiting temporal coherence and scene analysis. Similar attempts were discussed in [164], where road shape priors were introduced within a graph-cut scheme. An interesting yet effective consideration that may enhance the road surface detection stems from prior information. This support information, along

with a good localization, can be used to compensate the contradictory or treacherous situations that may arise from cluttered or compromised images. The authors in [165] suggest decoupling the segmentation process from the road-edge estimation using prior road models to safely navigate at intersection level. In [56], the authors propose to obtain road priors from the OpenStreetMap service.

One of the main disadvantages arising from the topological mapping services like OpenStreetMap consists in the coarse accuracy with respect to the road segments. Despite accurate statistics are not usually provided, nowadays these services can not provide high-levels of accuracy, *e.g.*, 10 cm or less [2]. Moreover, due to the collaborative nature of the project, together with a lack of automatic testing and validation procedures, the precision is not consistent within the database and, as an additional consideration, the alignment between the road graph and satellite imagery is not reliable in all areas.

Interesting approaches try to solve this problem by means of satellite imagery parsing. For an exhaustive survey of state-of-the-art approaches, which are usually related to the photogrammetry field, the reader is referred to [166]. Regarding the robotics and computer vision fields, former attempts addressed only the extraction of the road areas [167] although the most interesting works synergistically exploits mapping services or GIS projects. On the one hand, the authors in [59] propose to segment road regions leveraging aerial images and supervising the process using publicly available road vector data, but their approach does not update the GIS database. On the other hand, the authors in [60] propose to enhance the OpenStreetMap road graph by including information about road width and road segments centerlines. These enhancements are extremely valuable in the context of vehicle localization, since errors in road centerlines represent the most common problem in approaches that use this feature as the main clue to perform localization. The same authors extended their previous work including both aerial and ground imagery [168], introducing a fine-grained road semantics that includes lanes, sidewalks and parking lots. Pursuing lane-level localization objectives, the authors in [169] propose to exploit the objects present in the vehicle's surrounding and to describe the probabilistic dependencies between the object measurements by means of a factor graph model. Similar conclusions result from the work in [7], where Histogram of Oriented Gradients are used in order to align the images acquired from a front facing camera, and thus improve the localization of a vehicle.

Although the literature present a large number of contributions dealing with the interpretation of road images including approaches based on Markov Random Fields (MRF), Conditional Random Fields (CRF) or Convolutional Neural Networks (CNN), in the context of vehicle localization, additional high-level interpretations including, but not limited to, ego-lane detection may assist future navigation systems. For example, precise lane localization may help ADAS systems to recommend lane changing to human drivers. To deal with ego-lane estimation the authors in [170, 171] propose systems able to perform lane localization respectively in highway and urban scenarios, exploiting boosting classifiers and particle filtering approaches. A similar research was performed by [172], where a multiple evidence from a visual processing pipeline was combined within a Bayesian Network approach. With respect to the lane detection problem, the first pioneering approaches were achieved by Prof. Dickmanns [4], who exploits a 3D road representation model by means of clothoids curves and obtained with Kalman filters [173]. Besides color and texture, which are not always discriminative in terms of where the car can actually drive, road boundaries and lane markings are the main human perceptual clues [8]. These two elements may be detected using monocular vision, stereo vision, or even laser scanners sensors (leveraging reflectivity information). On the one hand, the main advantages of the laser scanners sensors stem from their active light source, which make the sensors less dependent from shadows and darkness issues, yet generating flawless distance measurements useful for curb and road shoulder detection. On the other hand, since road markings are designed to be visible by humans, vision algorithms, and in particular stereo systems, may take advantage from typical visual clues including different color-space transformations and 3D reconstruction capabilities. Unfortunately, considering that illumination issues as well as cluttering and marking irregularities could make the road marking identification hard, image pre-processing and road-model fitting procedures are required in order to achieve satisfying performances. Apart from these peculiar cases, vision-based approaches are nowadays cost-effective solutions in the automotive industry. For a comprehensive review of road and lane detection methods, the reader is referred to the excellent works in [8, 174, 175]. In this thesis we focus our attention on the number of lanes and the road width features, which are then matched against the information retrieved from the OpenStreetMap service. Moreover, we present an ad-hoc filtering scheme to handle the unavailability issue,

allowing us to refine the position of a vehicle with respect to a hypothesized on-road position.

2.9 Conclusions

In this chapter, we have presented a review of the state-of-the-art regarding scene understanding methodologies that can be used to perform robust vehicle localization, starting from the simpler robot-localization problem up to the integration of specific contextual clues usually available in driving scenarios.

Chapter 3

Ego Vehicle Localization

In this thesis, we address the problem of estimating a configuration of common road features (*e.g.*, buildings, intersections, road typology and related properties) within the perceived environment, by means of a flexible probabilistic framework for outdoor scene understanding. The configuration is then used to improve the ego-vehicle localization in typical driving scenarios, including urban areas and highway contexts, as depicted in Figure 3.1.

The proposed approach allows us to exploit a broad range of information sources, by incorporating them onto a common probabilistic basis. This range spans different physical sensor types (*e.g.*, cameras, laser scanners, GPS, proprioceptive sensors), and a disparate set of virtual sensors, *i.e.*, software components that can provide useful and pertinent information (*e.g.*, topometric maps, dimensions and appearance of buildings, etc.).

The chapter is organized as follows. Section 3.1 provides a detailed description of the proposed framework, which led to the publication of the work *A framework for outdoor urban environment estimation* [54]. An early working example of this approach is provided in Section 3.2, where an application to the vehicle localization problem in urban environments is suggested. So as to refine the achieved localization results, Section 3.3 discusses how to enhance the localization estimate by leveraging a pure-geometric building detector module, which was presented in *Leveraging the OSM Building Data to Enhance the Localization of an Urban Vehicle* [176]. Finally 3.4 reports how to integrate road properties like the with and the lane number inside the overall evaluation approach.

Figure 3.1: Example result using the proposed framework on run 2011_09_30_drive_0018 of the KITTI dataset, and the corresponding OpenStreetMap road network. The red path is the GPS-RTK and plays the role of ground truth. In green, the localization results of our framework.

3.1 The Road Layout Estimation Framework

Autonomous systems require an accurate understanding of the surrounding environment in order to safely plan their actions. For intelligent road vehicles one of such fundamental understanding concerns the pose of the vehicle, called localization. With respect to the literature mentioned in the previous chapter, the probabilistic framework proposed in this thesis presents some significant differences. On the one hand, other works focus on solving the localization problem leveraging a set of sensor types defined a priori, *i.e.*, at design time. On the other hand, our probabilistic framework is designed to be flexible so to allow us to exploit the information generated by any kind (physical or virtual), and any number of sensors, thus increasing the number of features contributing to theevaluation of the posterior probability of the vehicle pose. Our first claim is thus the easiness of changing the inference structure of the framework, *i.e.*, the type and number of the sensors and the geometric and semantic relationships between them in an on-line fashion. This pattern has been derived from the successful development for indoor scene reconstruction in [80] [81], although a complete generalization and extension of the approach is here introduced. From a technical perspective, the approach relies on the well-known bayesian particle filtering technique [39, Chapter 4], which allows us to keep track of a whole set of hypotheses, we

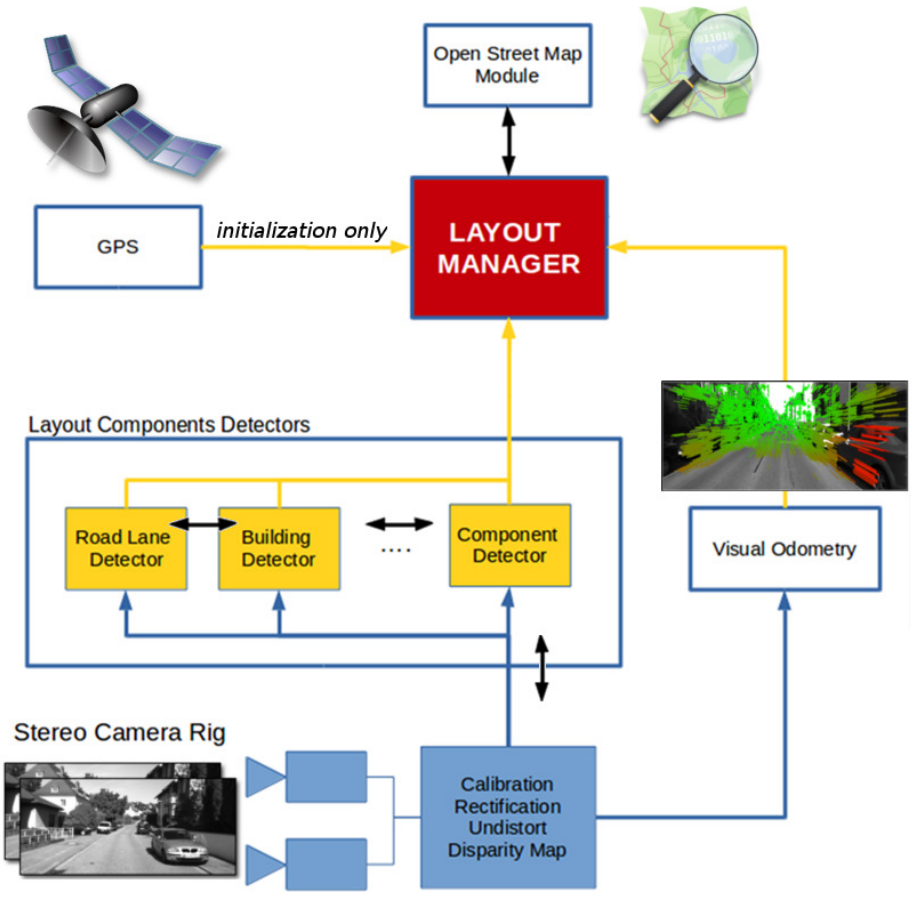


Figure 3.2: Proposed Road Layout Estimation Framework schematics. The inputs to our framework were the mainly sequences from the KITTI dataset and the associated OpenStreetMap road network. We also used the LIBViso2 [177] library to calculate the odometry displacements.

call *Layout Hypotheses* –LH–. This approach represents a step towards a more complex outdoor urban scene understanding system, meant to model the surrounding scene layout by means of *Layout Components* –LC–, each being generated by processing the sensors data with higher level detectors. The system enables us to approximate the posterior probability distribution, *i.e.*, a distribution over the possible configuration of the recognized features, which are then the model of the vehicle surroundings. In this thesis, we leverage the sensor fusion capabilities of the proposed system for the purpose of robust vehicle localization. Differently from the previously mentioned approaches for localization based on particle filters, here the particle states are enriched by layout components, in addition to the 6DoF pose coordinates. The main insight

behind the proposed architecture is that different information from different sources may be available at different frequencies, may be absent for some periods of time, and may even only be useful in specific situations. It follows that, in contrast to state-of-the-art methods where all the sensor outputs must be available to perform the evaluation, *e.g.*, they are all part of an a priori interaction scheme, in the proposed framework each sensor is allowed to contribute to the scene understanding process as its information becomes available. In other words, the importance factor of the particle filtering scheme takes into account only the available information, which is dynamically updated as sensor outputs are gathered by the framework. Conceptually, no sensor is actually essential, yet all concur to the improvement of the interpretation quality and accuracy, each sensor being allowed to both bring an independent contribution, as well as be part of an interaction scheme. The structure of the proposed framework is depicted in Figure 3.2.

In this chapter, we show the flexibility of the proposed Road Layout Estimation framework as applied to the localization problem in urban autonomous driving scenarios, synergically exploiting the information provided by physical and virtual sensors.

3.1.1 Layout Hypotheses

The core of the proposed framework relies on the Layout Hypothesis concept, which is a comprehensive representation of the vehicle state, including a 6DoF localization pose together with a classification of the surrounding scene. To comply with the probabilistic requirements, *i.e.*, to handle the sensor uncertainties, we propose a multi-hypotheses scheme in which every Layout Hypothesis corresponds to a candidate representation of the scene. In our work, given a whole set of Layout Hypothesis, we tackle the density estimation problem using a straightforward weighted average scheme, relying on the score of each Layout Hypothesis. Each Layout Hypothesis consists of the following structure:

- The vehicle state in terms of its 6DoF pose and its time derivatives.
- The vector of Layout Components, *i.e.*, the geometric/semantic models associated to scene elements.
- A value denoting the score of the layout, *i.e.*, an estimated value of the likelihood of the Layout Hypothesis, generated by a specific

scoring function that takes into account the likelihood of each single layout component as well as of their interactions.

- A motion model, which describes how the hypothesis evolve in time.

The Layout Components are the core of the scene understanding process, and they are essential for both the modularity of the scene representation and for the probabilistic evaluation of the Layout Hypothesis. In the context of intelligent road vehicles, typical scenes could present static elements like road markings, traffic signs and lights, buildings and crossings, etc., as well as dynamic objects such as pedestrians, other vehicles, or similar entities. The Layout Components come into life when detected by external modules, *i.e.*, implementations of machine perception algorithms, which process sensor streams to provide the component data to the system. Every instance of Layout Component lives independently from all the other Layout Components, whether being part of the same Layout Hypothesis or not. Its software implementation provide the functions to calculate both its own likelihood, *i.e.*, its contribution to the overall score of the scene, and the evolution of its state. The main advantages of the proposed choices are the following:

- The proposed structure gives a high degree of independence in the evaluation process. Since Layout Components are not required but they actively contribute to the overall score of the hypothesis, the evaluation process of the whole layout hypothesis could take into account only a subset of them.

- The particle filter approach allows us to handle multi-modal distributions. In the specific case of the localization problem it follows that we can draw samples from the whole state space, *i.e.*, the complete road graph of a city, and then efficiently track the hypotheses up to a correct localization that results from the detection of a new road feature not detected during the initialization phase.

- The logical separation of the detector from the Layout Components validation process allows us to exploit external detectors introducing high-level concepts within a Layout Component. As an example, a state-of-the-art line detector can be employed to probabilistically estimate the ego-lane within a multi-lane way.

- Even if the framework achieved near real-time performances in specific configurations, even on a single threaded machine, the designed architecture provides a straightforward way to speedup the overall system.

Finally, it is important to notice that the from a technical perspective the system could even be provided with randomly hypothesized components, if combinatorial explosion could be neglected. Nevertheless, it is clear that having a suitable object detector would allow the system to avoid an unnecessary waste of resources.

3.1.2 Hypotheses Initialization and Evolution

As observed from an autonomous vehicle in a typical driving scenario, the scene appearance changes both because of the vehicle motion and because of the dynamics of the moving objects in the scene.

Layout Hypotheses are generated grounding on the fair assumption that vehicles travel on roads. The hypotheses are allowed to evolve over time, to refine the quality of their initial description, and to contribute to the belief of the interpretation of the observed scene.

As a first step to cope with the time evolution, each Layout Hypothesis has to have means to estimate the motion with respect to the observed scene. Given the existence of different approaches to measuring the observer motion (*e.g.*, wheel odometers, visual odometers, etc.), we designed the framework so as to cope with different motion estimation sources. Each layout hypothesis has therefore an associated motion model, which integrates such measures into the hypothesis substate. Moreover, since external modules (*e.g.*, object detectors from images) might run orders of magnitude slower with respect to the update frequency of the Layout Hypotheses, all the Layout Components that cannot be updated, *i.e.*, those components that have not received a new input from the external detectors, are propagated taking into account both the observer motion, *i.e.*, the Layout Hypothesis motion, and the specific motion of each moving scene element, *i.e.*, the Layout Component motion. This expedient allows us to keep the update frequency independent of the detection modules, therefore preserving the real-time requirements.

3.1.3 Layout components

The vector of Layout Components associated to each Layout Hypothesis contains instances of the scene elements hypothesized by the external detectors. Since different typologies of scene elements provide higher level semantic interaction cues, the interpretation quality benefits from a rich and heterogeneous set of Layout Components, as that may help in disambiguating complex scenarios. The proposed framework offers several templates to easily integrate new scene components, and new ones can be straightforwardly defined. From a technical point of view, there are two main routines for each type of layout component software:

- propagateComponent: this routine composes the Layout Hypothesis motion with the component motion model, optionally adding a perturbation term to the motion. If a component does not have its own motion model, this function simply updates the component state using the information provided by the Layout Hypothesis, *i.e.*, uses the estimated observer motion of the hypothesis to generate a new component state. A detailed example is presented in Section 3.2.
- calculateComponentScore: this second routine evaluate the likelihood of the Layout Component instance given new evidence from the sensors.

3.1.4 Hypotheses evaluation

In the crucial step of evaluating the likelihood of Layout Hypothesis, each layout component contributes its own scoring term. The scoring terms of all the layout components within a layout hypothesis contribute to the final score of the hypothesis both individually and as part of an inter-component interaction scheme. This design choice enables the implementation of complex interaction schemes to cope with more complicated scenarios, allowing to exploit geometrical, physical and semantic constraints. As an example, the matching of the estimate of the road width with the size and position of a building façade against a topometric map, may considerably refine the quality of the scene layout. Furthermore, a Layout Hypothesis with a misplaced building Layout Component, *e.g.*, in the middle of a crossing area, will naturally turn extremely unlikely. After all hypotheses have been evaluated, the resampling step takes place. Here, a new set of hypotheses is drawn from the previously

evaluated Layout Hypothesis, taking in consideration their importance factor, *i.e.*, the score of the Layout Hypothesis.

The global scheme of the proposed framework is formalized in Algorithm 1.

Algorithm 1 Filter Layout Estimation

Require: $M \leftarrow$ surrounding map from OSM
$\forall\, l \in LayoutHypothesis \Rightarrow l_{pose} \leftarrow$ initialize with GPS
$\forall\, l \in LayoutHypothesis \Rightarrow l_{speed} = 0$

procedure FILTER($LayoutHypothesis_{t-1}$)
  if $new\ detection\ flag$ **then**
    for all $l \in LayoutHypothesis$ **do**
      $l.add(new\ detection)$
    end for
  end if
  for all $l \in LayoutHypothesis$ **do**
    $propagatePoseEstimation(l)$
    for all $c \in l$ **do**
      $propagateComponent$
      $calculateComponentScore$
    end for
    $calculateScore(l)$
  end for
  if $resampling\ interval\ reached$ **then**
    $resample\ hypotheses\ set$
  end if
  return: $LayoutHypothesis_t$
end procedure

3.2 Leveraging RLE for Vehicle Localization

In this section, we present an application of the proposed framework to the vehicle localization problem within urban driving scenarios, demonstrating the effectiveness by reaching state-of-the-art localization performances. For this purpose, we exploit the information provided by two physical sensors (a GPS receiver and a stereo camera) and one virtual sensor (a software module capable to retrieve information from a topometric map service). In this work, we use the road network obtained from the OpenStreetMap service, which provides open access to under the ODbL license [178], in contrast to the more restrictive Google Maps

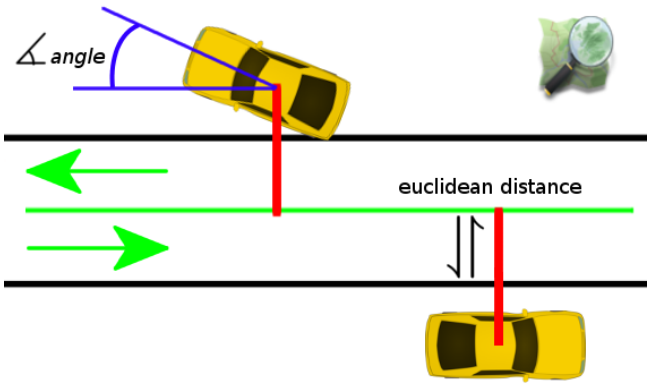


Figure 3.3: The OpenStreetMap Layout Component contributes to its Layout Hypothesis evaluation considering the distance (here represented with a red line) between the Layout Hypothesis position coordinates with respect to the nearest road segment (here represented with a green line) and the angular misalignment (here represented with a blue arc), which includes also the road driving direction.

terms of service [179]. The first step is a rough initialization of the system, in which the localization within a bounded area is assumed for all the Layout Hypotheses. For instance, we can use a circle of suitable radius, centered on the last known vehicle position or, if a GPS module is available, we can exploit the GPS fix to provide an initial, yet noisy, global latitude-longitude localization. Following the intuition provided in Section 3.1.2, this information will be used to generate a normally distributed set of layout hypotheses (in this case they are just localization hypotheses), in the area of uncertainty of the GPS estimate. The evolution of the Layout Hypotheses depends on the sensor readings and on the software components that process them. In this work, we propose to process the stereo camera stream using a state-of-the-art Visual Odometry approach, namely the LIBViso2 [177] library, to provide an estimate of the observer's motion. Our OpenStreetMap (OSM) software module (based on the Osmium Library [180]) automatically retrieves from the Internet the portion of map the observer is moving within and, by means of a map-matching technique, evaluates the error of each localization hypothesis with respect to the nearest road segment. As depicted Figure 3.3, this evaluation takes into account both the euclidean distance and the alignment error of the Layout Hypothesis pose with respect to the nearest OSM road segment, as well as the road driving direction.

Two out of three sensors will then behave as producers of very simple Layout Components, implementing the two required routines described in Section 3.1.3. In particular, the stereo camera component will propa-

gate according to the LIBViso2 [177] motion estimate and will provide a score reflecting the quality of its estimate, while the virtual sensor component (OpenStreetMap module) will propagate along the nearest map road segment. If a component does not have a proper motion model, like in the case of a GPS sensor component, it will simply integrate the information gathered by the sensor into its component state, allowing the framework to use the measurement during the Layout Component scoring step.

3.2.1 OpenStreetMap features

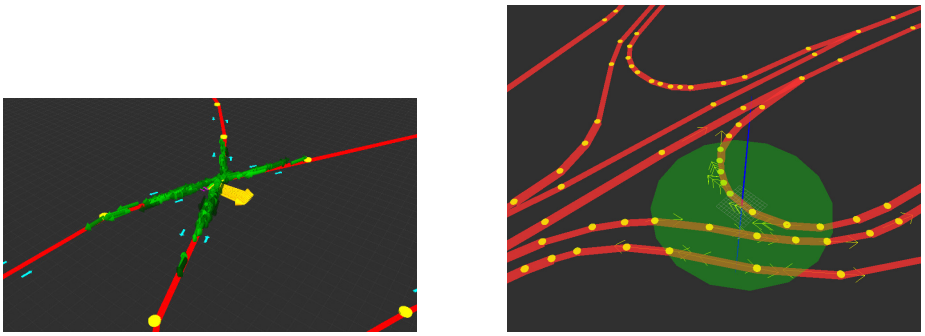
In the OpenStreetMap service[1], the map features are mainly composed of three elements, *i.e.*, *nodes*, *ways* and *relations*, to which several *tags* can be associated. The *nodes*, stored in WGS84 latitude/longitude coordinate system, represent the fundamental element for every other complex feature and therefore can be used to mark signs or, combining several points, to create the shape of a road. The *ways* are used to describe linear features such as road, roundabouts, building façades and other polylines, and usually are combined with *tags*, *e.g.*, street directions, number of lanes or road width. Finally, *relations* models logical and geographical relationships between objects, such as administrative boundaries.

3.2.2 OpenStreetMap module: Hypotheses initialization and Scoring Function

The OpenStreetMap module allows us to contribute to the estimate of a layout hypothesis, *i.e.*, its 6DoF pose, by means of the technique known in literature as *map-matching*, allowing the system to perform a *lock on road* [107] procedure. During the initialization step, initial poses of hypotheses are drawn from a normal distribution centered in the first available GPS information, with a standard deviation proportional to the GPS uncertainty. According to the constraint that cars are supposed to drive on roads, each hypothesis is roto-translated to the OSM module pose, which is guaranteed to lay on a road. Figure 3.4 depicts the initialization for the set of localization hypotheses.

During the hypotheses evaluation step, the scores are calculated according to the differences between the 6DoF poses and their projection,

[1] https://wiki.openstreetmap.org/wiki/Elements



(a) (b)

Figure 3.4: The figure depicts the initialization phase. The hypotheses, shown as green arrows, are drawn from a normal distribution, centered in the first available GPS position estimate, proportional to its uncertainty and snapped to the nearest road segment. In Figure 3.4a, the yellow arrows depict the LIBViso2 [177] odometry information (the vehicle axis orthogonal to the forward motion), and the road network is shown using red segments between OSM waypoints. We also plot the road driving directions using light blue arrows (close to the road segments).

snapped on the nearest OSM road segment (Figure 3.3), in terms of euclidean (Equation 3.1) and orientation (Equation 3.2) distances, considering also the road driving direction gathered from the OpenStreetMap module. The final score of the hypotheses is computed as in Equation 3.3, in which we also introduced two α-values which allows us to weight differently each contribution.

$$OSM_{=} = \frac{1}{\sigma_{=}\sqrt{2\pi}} e^{\frac{-\left(\mathrm{l}_{position} - \mathrm{snap}_{p}\right)^2}{2\sigma_{=}^2}} \tag{3.1}$$

$$OSM_{\measuredangle} = \frac{1}{\sigma_{\measuredangle}\sqrt{2\pi}} e^{\frac{-\left(\mathrm{l}_{rotation} - \mathrm{snap}_{R}\right)^2}{2\sigma_{\measuredangle}^2}} \tag{3.2}$$

$$LayoutHypothesis_i = \alpha_1 \cdot OSM_{=} \cdot \alpha_2 \cdot OSM_{\measuredangle} \tag{3.3}$$

3.3 The Building Model

In this section, we present a technique to detect and exploit building façades and the correponding OpenStreetMaps building outlines to improve the localization of a vehicle driving in an urban scenario. The proposed approach leverages images from a stereo rig mounted on the vehicle to produce a representation of the buildings' façades within the

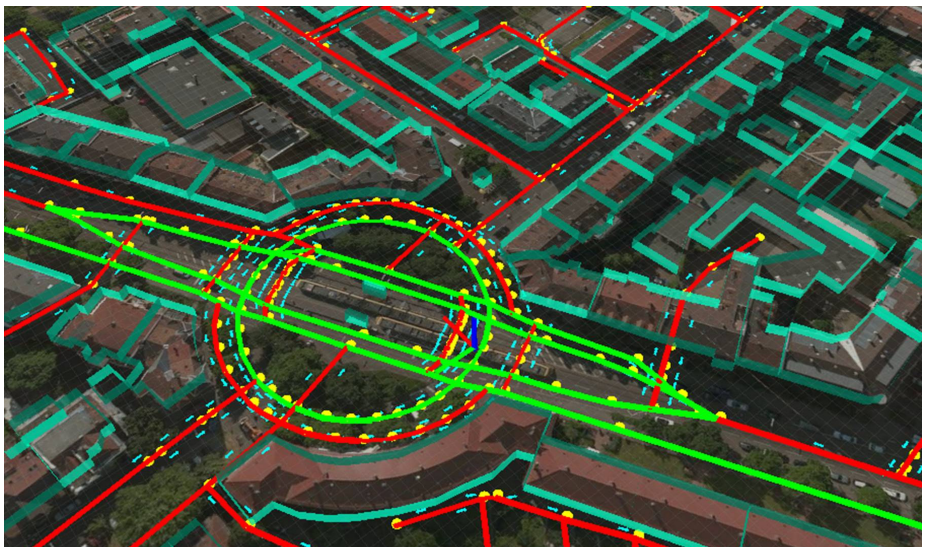


Figure 3.5: Example results from the proposed buildings approach on one sequence of the KITTI dataset, and the corresponding OpenStreetMap road network along with the building outlines. The green and red lines represent respectively the one-way and two-way traffic roads, while the buildings outlines are shown in pale green. The satellite image was also overlayed.

field of view. This representation is matched against the outlines of the surrounding buildings as they are available from the OpenStreetMap service. The information is then fed into our probabilistic framework as a new Layout Component, in order to produce an accurate lane-level localization of the vehicle. The experiments conducted on the well-known KITTI datasets prove the effectiveness of our approach.

3.3.1 Façades Detection Pipeline

In the context of urban areas navigation systems cannot rely on the GNSS signals (*e.g.*, GPS, GLONASS, Galileo) since it undergoes multi-paths and physical barriers, leading sporadically to very poor GNSS accuracy or even to no estimate at all. In order to overcome this issue, navigation modules usually couple the GNSS system with cartographic maps and methods that leverage the road graphs. These techniques are usually known as *lock-on-road* algorithms, see, *e.g.*, [9]. While these approaches led to remarkable increases in the localization accuracy, they yet do not allow us to achieve the necessary precision for a *lane level* localization, *i.e.*, accuracies in the order of $10cm$ [2]. However, we can extend the aforementioned approach by exploiting the buildings themselves, and

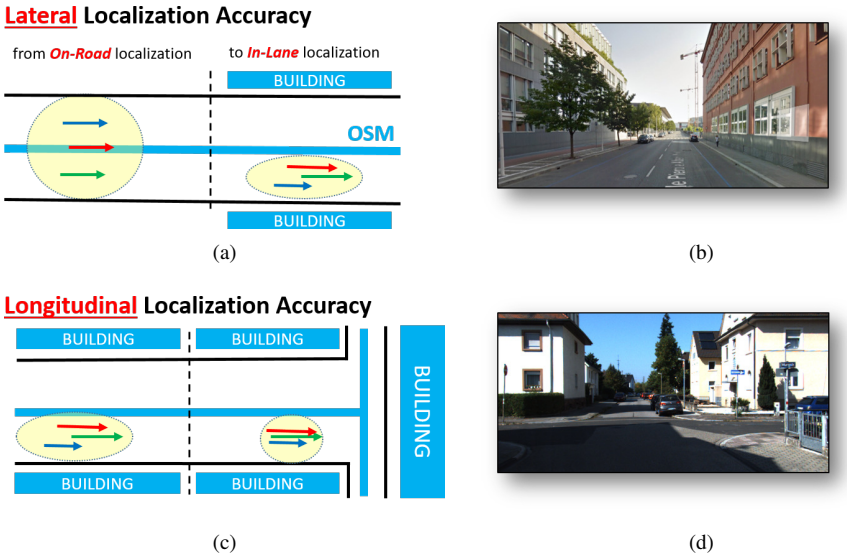


Figure 3.6: The figure depicts the localization accuracy in both the lateral and longitudinal directions. We aim to increase it by exploiting the building outline information from the OpenStreetMap service. Figures 3.6a and 3.6b refer/show a typical so-called "urban canyon situation", where the GPS signal may be degraded due to signal blockage and severe multipath. Figures 3.6c and 3.6c refer/show a residential area: in this case the buildings façades in the opposite side of the intersection may be employed to reduce the longitudinal localization uncertainty.

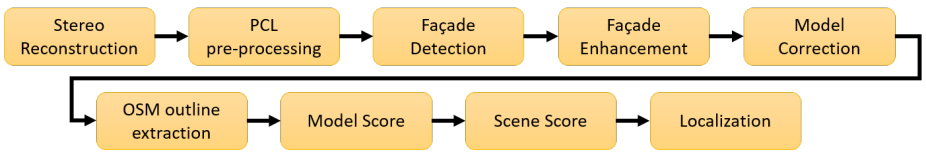


Figure 3.7: To generate a 3D model of the building façades within the scene, we developed the depicted detection pipeline, which bases on the images from a stereo rig.

integrate this new feature as a new Layout Component within the Road Layout Estimation framework. We model the building façades using the geometric pipeline depicted in Figure 3.7. In contrast to the complex yet time consuming techniques described in Chapter 2, our approach allow us to achieve a satisfactory, although not real-time, performance without having performed any specific code optimization. With respect to the localization setup proposed in Section 5.1.2, here the framework shows its effectiveness by means of the seamless integration of new components into the scene layout evaluation process.

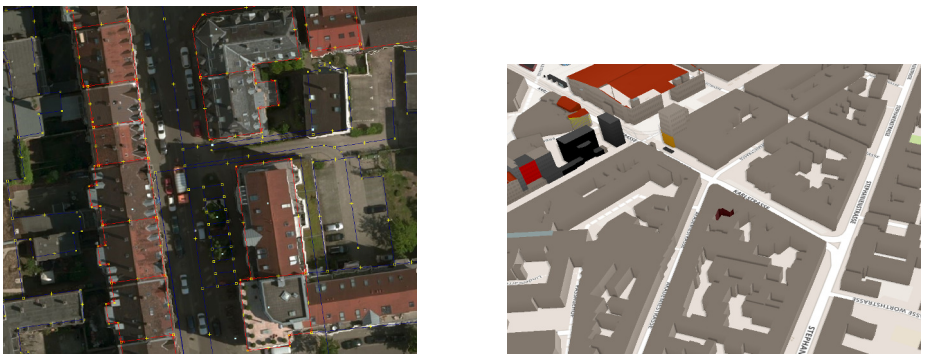
3.3.2 The Buildings Database

We propose to leverage the well-known OpenStreetMap service as information source for the building database. According to the OpenStreetMap data model, buildings' outlines can be retrieved selecting the *building* key from the list of ways. While the OpenStreetMap documentation refers to interesting additional tags, like the height of the building, we believe that, besides the *building* key, these tags are still not ready to be exploited, mainly because they are too rarely used, as shown in Figure 3.8b. After a rough preliminary analysis of the height of the buildings in our testbed dataset, we choose to set the building height to a fixed value, but, thanks to the nature of the proposed approach, this coarse approximation does not introduce any bias into the localization algorithm. While the OpenStreetMap service does not guarantee any accuracy about the outlines of the buildings, the accuracy of their position, at least in the KITTI datasets that we could inspect, is usually higher than one meter. In other words, their positioning is accurate enough to potentially allow us to gain a lane-level lateral localization accuracy as depicted in Figure 3.6a. As it is shown in Figure 3.8a, the outlines are almost always in overlap with the real shape of buildings.

3.3.3 Detecting the Building Geometry from image data

To extract a geometric model of the surrounding buildings, the first step of our pipeline consists in extracting the related façades from the sensor images. We leverage the Semi-Global Block Matching algorithm [181] (SGBM), available in the OpenCV library[2], a well-known 3D stereo reconstruction pipeline. This allows us to retrieve a 3D point cloud

[2] http://opencv.org/



(a) (b)

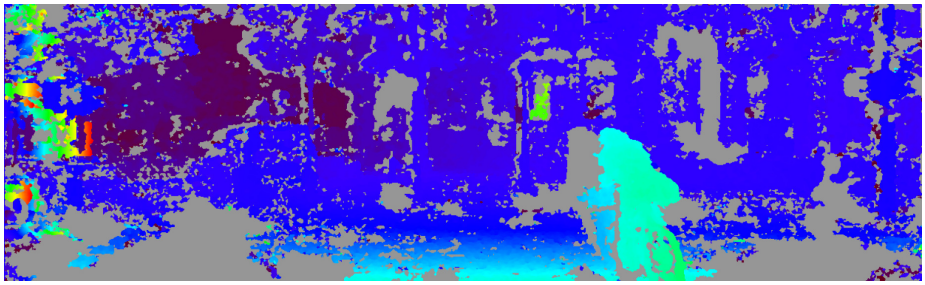
Figure 3.8: An example of cartographic map gathered from the OpenStreetmaps service. In Figure 3.8a the red boxes around the buildings represent the outlines provided by the service. In Figure 3.8b a 3D view of the buildings data from the osm-buildings.org website clearly shows the great number of buildings without a populated height tag (middle 2016).

from the stereo camera stream, as shown in Figure 3.13a. An ad-hoc parametrization of the SGBM algorithm was required to obtain useful results; it mainly differs from the original parametrization in the *Correlation Window Size* and *P1/P2* parameters, which were adapted so as to better perform with the façade detection. An example of the achievable results are shown in Figure 3.9 and Figure 3.10.

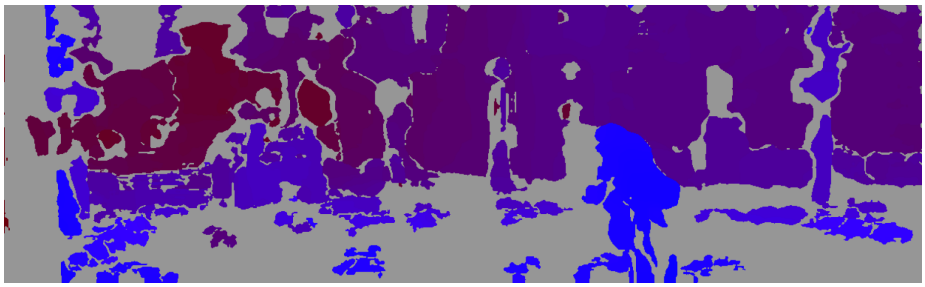
Furthermore, a refining phase is performed as follows. We first remove every 3D point that does not lie inside a 3D bounding box representing the visible surroundings with acceptable tolerance. We set the limits according to the field of view of the camera, resulting in $50m$ with respect to the longitudinal axis, $\pm 50m$ on the lateral axis and a $[-0.4, 16]m$ threshold with respect to the vertical axis. Given the KITTI stereo camera configuration, the resulting bounding box cuts objects under $1.25m$ with respect to the flat road plane. Conceptually, this expedient allows us to filter the most cluttered area in the field of view, yet keeping the main structure of the façades untouched, as depicted in Figure 3.13b.

After this preliminary stage, we calculate the surface normals of the resulting point cloud. The insight here is to keep only the points lying on similar planes, *i.e.*, removing the outliers not staying on same plane before the following filtering step. In order to calculate the normals vectors, we used the Integral Images method proposed by [182] and implemented in the PCL Library, achieving very good results and performance speeds comparable with the KITTI dataset frame rates, *e.g.*,

(a)

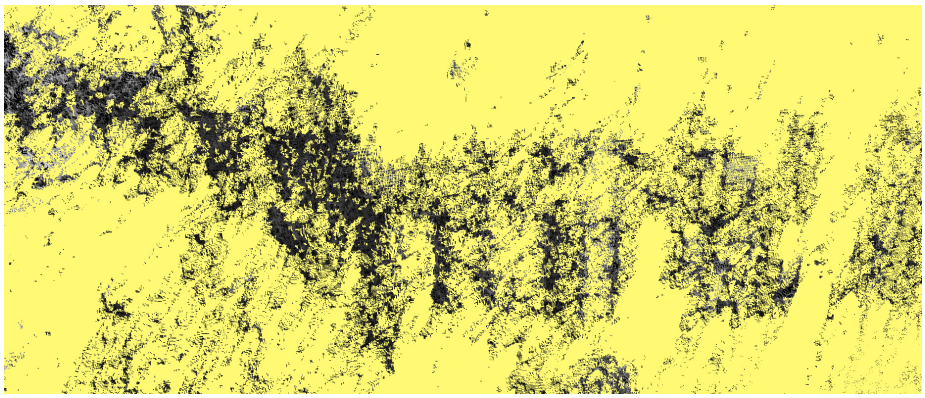


(b)



(c)

Figure 3.9: In Figure 3.9a, a frame from the KITTI dataset, sequence 2011_09_26_drive_0005, and the corresponding depth map as obtained with the standard parametrization of the SGBM algorithm in Figure 3.9b. Our custom parametrization, depicted in Figure 3.9c aims to maximize the façade detection.



(a)

(b)

Figure 3.10: Resulting 3D point clouds obtained using the standard SGBM parametrization (Figure 3.10a) and our parametrization (Figure 3.10b).

(a) Output of the Region Growing Segmentation algorithm. The area on the middle contains 3 hidden façades

(b) In the picture, the desired segmentation is shown.

Figure 3.11: The figure depicts a typical segmentation error. Using only the angle between normals and the surface curvature obtained from the 3D point cloud only, the Region Growing Segmentation algorithm detected only 3 (Figure 3.11a) out of 5 façades (Figure 3.11b) present in the segmented areas.

$10Hz$. To further refine the results, we use a conditional filter over the normal vector associated with candidate point. The assumption is that building façades are almost orthogonal to the driving plane, and, to better cope with the reconstruction noise, a threshold is applied according to Equation (3.4), where $n = [n_x, n_y, n_z]^T$ is the normal vector associated with point p and th represents the desired threshold. In this work, we chose to keep the points under a threshold value of $20°deg$.

$$|cos(\alpha)| = \frac{|n_z|}{\sqrt{n_x^2 + n_y^2 + n_z^2}} < cos(\frac{\pi}{2} - th) \tag{3.4}$$

3.3.4 Semantic Segmentation

After the first filtering step, during which the resulting point cloud is processed to reduce the amount of clutter, a two-phase façades recognition takes place. The first step involves a clustering procedure by means of the Region Growing Segmentation algorithm implemented within the PCL library. This algorithm aims at merging points that are considered to lie on the same plane when the angle between their normals is below a given threshold. In this way we create a cluster of points lying on the same smooth surface. The main insight from this procedure is to further improve the façade detection, leveraging the smoothness of the surfaces. The parameters θ_{th} and c_{th} of the Region Growing algorithm, representing the *angle between normals* and the *curvature* threshold for the surface analysis, are respectively set to $10°$ and 0.1. The results of this first steps are depicted in Figure 3.13c.

Although this procedure is usually able to identify the façades rejecting the most common sources of noise, it is not the only procedure used during the segmentation phase. Multiple consecutive planes yet not be-

longing to the same building might be detected inside a single cluster, as depicted in Figure 3.11. Recalling that the previous phase is dedicated to enhancing the point-to-façade affinity rather than associating points to a specific façade, the following step aims at extracting the planar region patches association for each point inside each cluster. As we cannot easily know how many planes are available inside each cluster, we used the iterative Plane Model Segmentation algorithm of the PCL libraries in order to cope with an arbitrary number of planes. The method consists in evaluating the points inside each cluster, using a distance threshold that refers to the standard plane equation. As the number of points belonging to each plane cannot be specified a priori, the procedure consecutively performs a search phase in order to detect planes orthogonal to the road surface. The procedure grounds on the RANSAC approach and exits when the number of points, *i.e.*, the outliers, fall below a 10% threshold over the initial number of points. The number of RANSAC iterations was evaluated using Equation (3.5) [183], where p value was set to 0.99 and m value equals to 3 since three non-collinear points are needed to estimate a plane equation. According to our experimental activities, we achieved a good plane detection using $v = \frac{3}{5}$ and thus an average of 70 RANSAC minimum iterations.

$$N = \frac{log(1-p)}{log(1-(1-v)^m)} = \frac{log(1-0.99)}{log(1-(1-0.6)^3)} = 69.63 \approx 70 \quad (3.5)$$

3.3.5 Post Processing

The results after the aforementioned steps consists of a set of candidate façades, represented by plane equations. Since we gathered the 3D façades from the building database using a simple extrusion, *i.e.*, introducing an orthogonality constraint with respect to the road surface, we need a perpendicularity correction of candidate plane equations before performing the evaluation step. On the one hand, the thresholds applied to the RANSAC procedure clearly lead to an approximation. On the other hand, the presence of architectural structures, like windows or balconies, limit the accuracy of the proposed pipeline as a whole façade is always approximated by a single 3D plane or patch. However, despite this apparently rough estimate, we verified that these detected façades are extremely useful to achieve a good enhancement in terms of lane-level localization. We apply the perpendicularity correction by simply

(a) (b) (c)

Figure 3.12: Perpendicularity correction. In Figure 3.12a blue color represents the original plane and red the plane made orthogonal to the ground plane. Figures 3.12b and 3.12c depict a typical façade with a balcony, which introduces a bias in the plane estimation using our pure geometric approach.

replacing the candidate façades with new planes oriented as the original ones, but orthogonal to the ground plane, as depicted in Figure 3.12a.

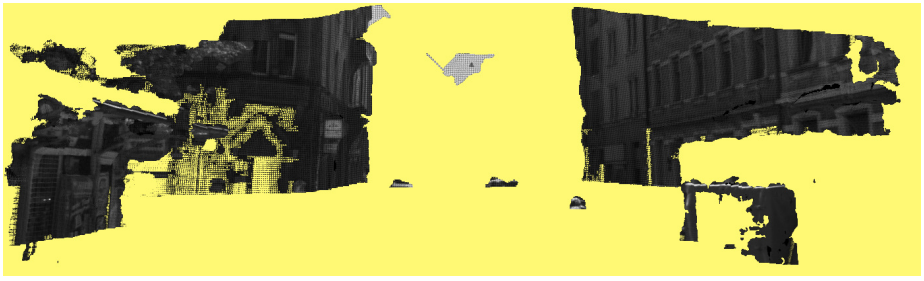
The results up to this phase are then a set of plane equations, representing the candidate façades detected from the stereo image pair.

After a new pair of images is received and processed, the next step of our pipeline is to query all buildings outlines within an area of a given threshold (st to $30m$) centered on the vehicle position stored within the Layout Hypotheses. We then build the 3D model of the surrounding buildings, *i.e.*, an approximation of the building façades by means of plane equations extruded orthogonally to the road surface. Figure 3.5 depicts the resulting output of this step.

3.3.6 Layout Component Scoring function

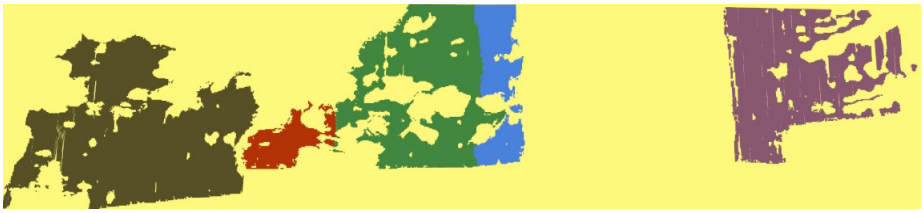
The previous sections proposed a façade detection pipeline. To exploit the results within the Road Layout Estimation framework we need to introduce a new Layout Component, *i.e.*, the Building Component, which contains instances of the scene elements hypothesized by the aforementioned detection scheme. To allow the framework to enhance the localization accuracy using this information, the Layout Component needs to implement the *calculateComponentScore* routine described in Section 3.1.3 (since the buildings are not supposed to move, the *propagateComponent* routine is absent). Although all planes from both the detection pipeline and the OSM service are described as full 3D planes, we exploited the orthogonality to the road constraint in order to base on a simpler 2D scoring function. Recalling that this module aims at enhancing a localization estimate, we evaluate the detections as follows. First of all, the points representing a building façade are transformed with re-

(a)



(b)

(c)



(d)

Figure 3.13: The figure depicts the steps of the pipeline. The yellow color represents the unreconstructed areas, *i.e.*, the parts that will not be used in the reconstruction. In Figure 3.13a, the point cloud from the SGMB phase, in Figure 3.13b, the same image after the bounding box application. Please note that the façade structures are almost preserved. Figure 3.13c depicts the results of the Region Growing segmentation algorithm, while in Figure 3.13d the results after the final RANSAC iteration.

spect to the vehicle position, *i.e.*, the hypothesized position we might have to enhance. From the OpenStreetMap endpoints of the segments describing the building outlines, we define an edge e as follows:

$$e =< A, B, M, \epsilon > \tag{3.6}$$

where:

- $\mathbf{A} = [x_A, y_A, z_A]^\intercal$ and $\mathbf{B} = [x_B, y_B, z_B]^\intercal$ are the endpoints of each segment of the building outline list from OpenStreetMap.
- $\mathbf{M} = [x_M, y_M, z_M]^\intercal$ is the average point of the segment of building outline.
- Let $\epsilon = [a_\epsilon, b_\epsilon, c_\epsilon, d_\epsilon]^\intercal$ be the plane perpendicular to the road ground plane and passing through $\mathbf{A}$ and $\mathbf{B}$. Then let $\mathbf{u} = \mathbf{B} - \mathbf{A}$. Then define:
  - $[a_\epsilon, b_\epsilon, c_\epsilon]^\intercal = \mathbf{u} \times \mathbf{z}$.
  - $d_\epsilon = -(a_\epsilon x_A + b_\epsilon y_A + c_\epsilon z_A)$.

We further define a *façade* as:

$$f =< P, \pi, C, score : (F \times E) \rightarrow [0; 1] > \tag{3.7}$$

where:

- P is the set of points that belong to a plane
- $\pi = [a_\pi, b_\pi, c_\pi, d_\pi]^\intercal$ is the façade plane model
- C is a subset of edges of the OpenStreetMap map E. The items in C are those candidates that match the bounding box defined in Section 3.3.3.
- $score$ is the mapping function between *edges* and *façades*

Thus, for each detected *façade* we consider all of its close *candidate edges* in the OpenStreetMap map. The *score* is calculated by evaluating the following geometric relationships:

- A Cartesian distance between the points belonging to the *façade*, *i.e.*, the remaining points after the detection pipeline, with respect to the *candidate edges* as in Equation (3.13)

- A measure regarding the edge-to-façade misalignment, as in Equation (3.14).

From a technical perspective, the better the alignment between the detected façade and the edge calculated using the OpenStreetMap data, the higher the score relative to the vehicle position is. We hypothesized a normal distribution for each of the previous distances, to individually evaluate each score, using Equations (3.8) to (3.11).

$$f(\bar{d} \mid \mu_d, \sigma_d) = \frac{1}{\sigma_d \sqrt{2\pi}} \cdot e^{\frac{-(\bar{d} - \mu_d)^2}{2\sigma_d^2}} \tag{3.8}$$

$$f(\alpha \mid \mu_\alpha, \sigma_\alpha) = \frac{1}{\sigma_\alpha \sqrt{2\pi}} \cdot e^{\frac{-(\alpha - \mu_\alpha)^2}{2\sigma_\alpha^2}} \tag{3.9}$$

$$score(\bar{d}) = \frac{f(\bar{d} \mid 0, \sigma_d)}{f(0 \mid 0, \sigma_d)} \tag{3.10}$$

$$score(\alpha) = \frac{f(\alpha \mid 0, \sigma_\alpha)}{f(0 \mid 0, \sigma_\alpha)} \tag{3.11}$$

$$score(f_i, e_j) = c_1 * score(\bar{d}) + c_2 * score(\alpha) \tag{3.12}$$

Given these two scores, which are combined as in Equation (3.12), the overall façade-to-OpenStreetMap plane association is then performed employing a *winner-takes-all* strategy as described in Algorithm 2. Conceptually, we calculate the score with respect to all the candidate edges, then returning the highest achieved score.

Finally, the overall scene score is evaluated considering the average score of all the façades as in Algorithm 3, weighted using the number of points associated with each façade as show in Equation (3.15). This expedient allows to tailor the weighting scheme of the façades using the most likely ones, as depicted in Figure 3.14.

$$\bar{d} = \frac{1}{|P|} \sum_{i=1}^{|P|} \frac{|a_\epsilon x_i + b_\epsilon y_i + c_\epsilon z_i + d_\epsilon|}{\sqrt{a_\epsilon^2 + b_\epsilon^2 + c_\epsilon^2}} \tag{3.13}$$

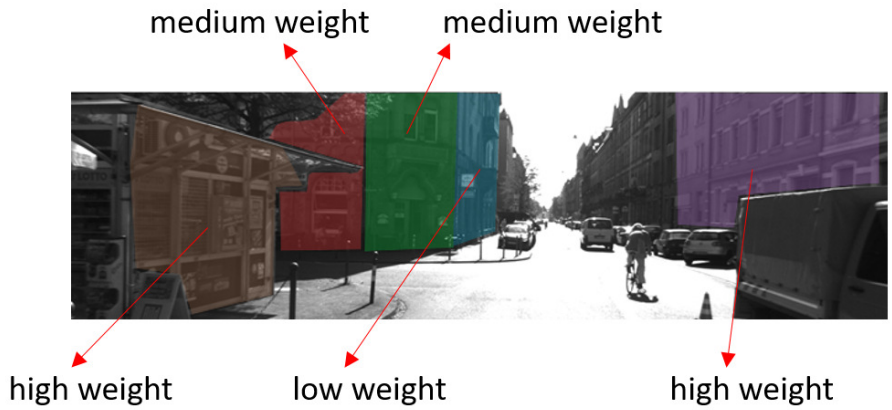


Figure 3.14: The picture shows how the façade are differently weighted with respect to the numer of associated points. The idea here is to give higher scores to the façades better detected.

$$\alpha = \arccos\left(\frac{a_\pi a_\epsilon + b_\pi b_\epsilon + c_\pi c_\epsilon}{\sqrt{a_\pi^2 + b_\pi^2 + c_\pi^2}\sqrt{a_\epsilon^2 + b_\epsilon^2 + c_\epsilon^2}}\right) \tag{3.14}$$

$$scene\_score = \frac{\sum_{i=1}^{F} |P_i| \cdot score(f_i)}{\sum_{i=1}^{F} |P_i|} \tag{3.15}$$

An overview of the pipeline steps is shown in Figure 3.7.

Algorithm 2 scoreCandidates

Require:
$f =$ the façade to score
$E =$ OpenStreetMap edge set
$thresh =$ proximity threshold

```
1: C ← ∅
2: for e in E do
3:     midpoint ← e.M
4:     for p in f.P do
5:         dist ← dist(p, midpoint)
6:         if dist < thresh then
7:             C ← C ∪ e
8:             break
9:         end if
10:    end for
11: end for
12: return C
```

Algorithm 3 Overall Scene Score

Require:
$F =$ segmented façade set
$E =$ OpenStreetMap edge set

```
1: scene_score ← 0.0
2: norm_term ← 0
3: for f in F do
4:     scores ← ∅
5:     f.C ← findCandidates
6:     for c in f.C do
7:         scores ← scores ∪ score(f, c) //Equation (3.12)
8:     end for
9:     scene_score ← scene_score + f.|P| × max scores
10:    norm_term ← norm_term + f.|P|
11: end for
12: scene_score ← scene_score/norm_term
13: return scene_score
```

3.4 Road markings: Width and Lanes

The aforementioned building component brings major benefits in urban or residential areas, where the building façades are usually clearly visible in the image. A completely different driving scenario is represented by the highway environments. As opposed to the previous environmental conditions, here the framework configuration cannot rely on architecture elements, thus returning to the cartographic only situation described in Section 3.2. It is worth mentioning that, despite the absence of huge obstacles and even in open-space areas, the GNSS signals alone cannot provide the adequate accuracy for an in-lane localization. In this section, we provide two methods designed to enhance the localization in typical highway environments, by leveraging the road width and the road number of lanes retrieved from the OpenStreetMap module. Both systems rely on a very simple road line tracker that was kindly provided by the INVETT Research Group (previously ISIS Research Group) of the Universidad de Alcalá, Alcalá de Henares - Madrid, lead by the Professor Miguel Ángel Sotelo Vázquez. The problem of line and lane detector has been tackled since the first approaches of the Professor Ernst Dickmanns in the mid of the 80's. Since then, a considerable amount of scientific work led to great advantages in [3, 4, 5, 6] and we refer the reader to the work in [8] for a comprehensive survey of the state-of-the-art approaches.

Instead of developing a new line detector, the idea behind the next two Layout Components is to exploit the available road properties and, leveraging the modularity of the proposed framework, reduce the localization uncertainties.

3.4.1 Line detector

In this section, we shortly describe the line detection and tracking algorithm used in this thesis. The purpose of this presentation is to highlight the pros and the cons of the approach, which are addressed by the proposed enhancement schemes. From a technical perspective, the pipeline leverages the image information of an on-board stereo rig, with known calibration with respect to the vehicle's reference frame. The algorithm consists of the following steps, which are also depicted in Figure 3.15:

- A first image processing phase, aimed at extract the contours of the road markings, is performed in the Bird Eye View (BEV) of the

right camera. For this reason, an homography matrix is computed, leveraging the intrinsic values of the camera and the extrinsic values with respect to the road surface.

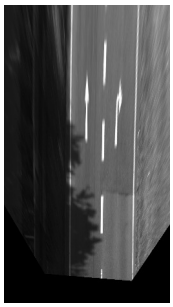
- The BEV image is then evaluated leveraging the algorithm proposed in [184].
- From the contours image, the white areas are evaluated considering their size. All areas below a parametrized threshold are discarded.
- At this time, the stereo image is exploited. Considering one contour at time, the algorithm tries to model the corresponding lines by fitting a clothoid model, considering the height value associated to each point of the image. To perform the latter 3D evaluation, the system leverages the SGBM algorithm in order to estimate the road plane equation. Differently from the original INVETT algorithm's, in our implementation we have exploited the more efficient ELAS [185] algorithm.
- According to the 3D road plane, the clothoids are then evaluated in order to prune out false detections.
- The retrieved clothoids parameters are then processed by means of a Kalman Filter that allow the detector to track the lines in time. A hysteresis counting procedure is also used to track the reliability with respect to the last 10 frames.

With respect to the performance evaluation of this simply algorithm, it has to be said that the algorithm achieves good performances only under optimal illumination conditions. As the reader may notice in Figure 3.16 dashed lines and shadows are not always handled correctly. However, this simple and, to some extent, naive detector, allows us to efficiently evaluate our next Layout Components, which are specifically designed to enhance the vehicle localization estimate by exploiting a noisy sensor, and the *width* and *lanes* values stored in OpenStreetMap road.

3.4.2 Road Width Component

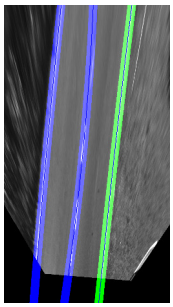
The first and simpler road component we introduce, tackles a common issue that arises with lock-on-road localization algorithms such as in Section 3.2. Consider for instance the situation depicted in Figure 3.17, where a vehicle that has been traveling in the rightmost lane for a while

(a) A frame from the KITTI dataset

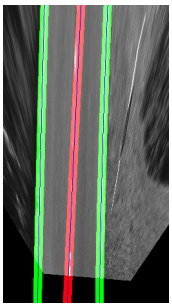


(a) Bird Eye View (BEV)

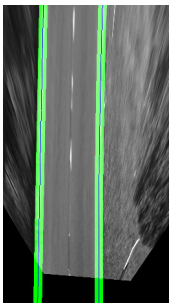
(b) Contours in BEV



(c) Tracking Example 1



(d) Tracking Example 2



(e) Dashed line issue

Figure 3.15: INVETT Line Tracker color code: green, clothoids currently tracked; red, clothoid model without support in the current image; blue, new clothoid, not yet tracked. Images from the KITTI dataset, sequence 2011_10_03_drive_0042.

turns onto the exit ramp that runs along the main highway. In this case, relying on a lock-to-road procedure only may easily lead to catastrophic errors, since the vehicle dynamics and the GNSS insufficient accuracies may not allow the vehicle to quickly recover from a complete wrong localization estimate. A comparable situation occurs when two parallel roads are in close proximity. In this section, we propose to tackle this issue by leveraging the road width information calculated using the aforementioned line detector, and the width tag provided by OpenStreetMap. In particular, the Road Layout Estimation framework will be exploited by means of a new Layout Component, aimed at improving the localization estimate in parallel roads contexts.

Width Estimation

To evaluate the road width measure, we modified the original INVETT line detector algorithm in order to output a new *is-continuous* flag associated with each detected line (this was done by trivially counting the number of white pixels over the whole clothoid area). Let us consider the example in Figure 3.18a, in which the output of the detector will be as in Figure 3.18b. From these values we calculate the road width by

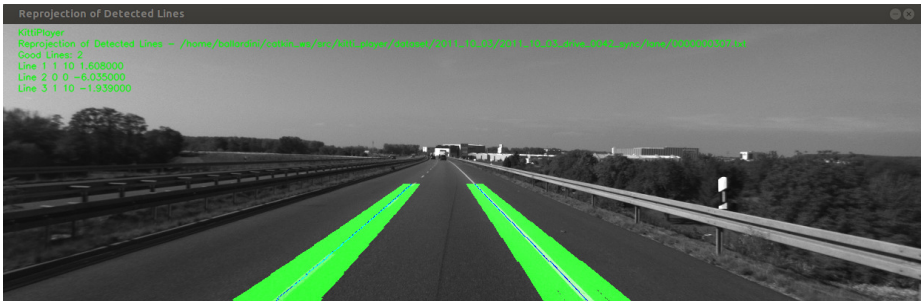
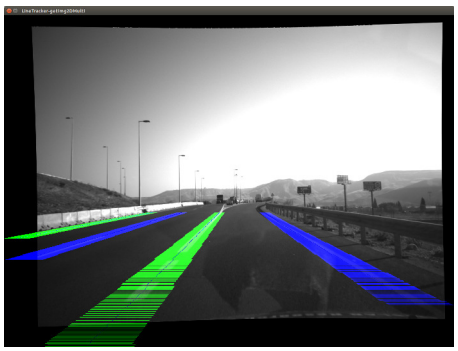


Figure 3.16: In this figure, two out of three lines are correctly tracked. The shadow created prevents the correct detection of the third line. Comparable issue arises also with dashed lines, if the space between two consecutive dashes is too large. An example of this problem is depicted in Figure 3.15e. In the image, the distances with respect to the lines are overlayed together with the ten-frames counter used to evaluate the line reliability. In the overlay, the Line 1 is valid (1), it has been correctly tracked in the last (10) frames and it is (+1.60m) far from us (on the right).

Figure 3.17: The figure depicts a common highway scenario with two parallel roads. In these cases, a simple map matching algorithm is not able to discriminate the correct vehicle position.



(a)

```
lines:
  -                                 -
    isValid: True                     isValid: True
    continuous: True                  continuous: False
    counter: 10                       counter: 0
    offset: -9.1509718895             offset: 0.99171652794
  -                                 -
    isValid: False                    isValid: False
    continuous: False                 continuous: True
    counter: 9                        counter: 7
    offset: -6.47675037384            offset: -2.11813640594
```

(b)

Figure 3.18: In the figure, the detected lines are re-projected in the 3D image-space. The corresponding line measurements are depicted on right.

evaluation one of the following two measures.

- If at least one continuous line is detected, we calculate the *DetectedWidth* measure considering the farthest continuous lines from each side (if only one is available, then we consider this measure as the full road width). Please notice that we consider the continuous lines as the road boundaries.
- Otherwise we calculate the *DetectedNaiveWidth* measure, considering the closest detected lines from each side, even if not valid, multiplied by lane number of the current road.

Following the same pattern as described in the previous sections, we introduce a new Layout Component, which is composed of a one single value, that is, the road width. During the initialization phase of the component, we exploit our the OpenStreetMap module to retrieve the nearest road segment and its associated width attribute. If the width value is not available, then we evaluate the width leveraging the more common *lanes number* tag, multiplied by the standard width value of 3.75 m (or according to the national traffic code). We model the two fundamental Layout Component routines as follows. With regards to the first *propagateComponent* routine, here we introduce a small perturbation term that allow us to handle the OpenStreetMap approximations. The predicted width value is sampled from a mixture distribution that is composed of two normals, respectively:

- $\mathcal{N}_1(\mu = \text{LayoutComponentState}_{t-1},\ \sigma^2 = \sigma_1)$
- $\mathcal{N}_2(\mu = \{\text{DetectedNaiveWidth or DetectedWidth}\},\ \sigma^2 = \sigma_2)$

Table 3.1: Mixture Weighing Scheme

	$\text{Width}_{\text{t-1}}$	CurrentWidth
DetectedWidth	0.1	0.9
DetectedNaiveWidth	0.3	0.7
No Lines Detected	0.9	0.1

Please notice that the μ value of the $\mathcal{N}_2$ distribution is evaluated using the appropriate width, *e.g.*, one of the two aforementioned widths (depending on the detector's output). Finally, if no line is found, accordingly to the national traffic code, the mixture varies as follows:

- $\mathcal{N}_1(\mu = \text{LayoutComponentState}_{t-1},\ \sigma^2 = \sigma_1)$
- $\mathcal{N}_3(\mu = 3.75\text{m} \cdot \text{Lanes Number},\ \sigma^2 = \sigma_3)$

In both cases, the mixture components are weighted using the scheme given in Table 3.1. Please note that the $\text{Width}_{\text{t-1}}$ and the CurrentWidth represents respectively the previous Layout Component state and the sensor reading. An ideal line detector would always measure the correct road with. Unfortunately, our detector is far from the perfection. Following the probabilistic approach of our framework, we explicitly model the sensor noise by means of a simple measurement model. We incorporate the "missing line" error by introducing a further mixture of two new densities, which represent our estimation error, *i.e.*, a wrong width estimate due to a line missing. The mixture is composed as follows and the weighted average is computed using the parameters in Table 3.2.

- $\mathcal{N}_4(\text{Expected OSM width + 1} \cdot \text{Standard Lane Width},\ \sigma^2 = \sigma_4)$
- $\mathcal{N}_5(\text{Expected OSM width },\ \sigma^2 = \sigma_5)$
- $\mathcal{N}_6(\text{Expected OSM width - 1} \cdot \text{Standard Lane Width},\ \sigma^2 = \sigma_6)$

At this point, the framework executes the second *calculateComponentScore* routine of the Layout Component. This second step is designed to incorporate the sensor measure into the previously evaluated *prediction*. From a theoretical point of view it represents the *measurement update* step of a generic Bayes Filter [39].

3.4.3 Lane Component

The second component we introduce consists in a lane detection module. This component is designed to increase the in-road localization ac-

Table 3.2: Sensor model Mixture Weighing Scheme

	Weighing Parameter
Weight for OSM width + 1	0.2
Weight for OSM width	0.6
Weight for OSM width - 1	0.2

Figure 3.19: An example of overexposed frame from the KITTI dataset, sequence 2011_10_03_drive_0042.

curacy of the OpenStreetMap scheme proposed in Section 3.2, allowing the framework to achieve in-lane localization in highway scenarios. On the one hand, the understanding of the vehicle's ego-lane can be considered as a by-product of the line detection procedure. In fact, having the relative positions of all the road lines within the road may allow us to simple evaluate the current lane using some trivial geometric considerations, in a per-frame basis. Unfortunately, a reliable full-line detection is usually hampered by vanished lines, cluttering elements or weather conditions as in Figure 3.19 and Section 2.8.3. On the other hand, let us consider the situation depicted in Figure 3.20a, which is surely a critical situation concerning the ego-lane identification problem. Even though no exact positioning can be estimated from the single detection shown in the image, a distance measure from the lane would enable us to limit the uncertainties to the compatible lanes only, as depicted in Figure 3.20b. Our proposal is then to tackle the ego-lane estimation by exploiting a probabilistic approach, in order to allow the system to infer the ego-lane estimation by leveraging consecutive, yet incomplete observations over the time. From a technical perspective, we opt for a Hidden Markov Model approach [186] with n-lane states, corresponding to the number of traffic lanes as stored within the OpenStreetMap service.

Figure 3.20: In this situation, only one out of four lines are detected. In Figure 3.20b the highlighted lanes correspond to the higher probability of the vehicle ego-lane, evaluated by leveraging the relative distance with respect to the only detected line.

3.4.4 Final considerations

In this last section we proposed two algorithms aimed at enhance the initial vehicle position provided by the lock-on-road procedure shown in Section 3.2. We took into account two specific features of typical highway scenarios, *i.e.*, the width and the number of lanes, together with a basic line detector that allowed us to verify our intuitions. Despite the quite simple considerations introduced, the experimental results in highway environments demonstrate the effectiveness of this approach.

3.5 Conclusions

In this chapter we have presented the Road Layout Estimation framework, together with three Layout Components and the associated sensing pipelines that allowed us to tackle the main challenges in the context of vehicle localization. The proposed framework has proved to be effective, allowing a seamless integration of new components into the localization process.

Chapter 4

Intersection Detector

In this chapter, we present an Intersection Detector module that aims to recognize the geometric configuration of road crossings. The discussion of this new Layout Component is separated from the previous components because of two reasons. First, while retaining an on-line approach as the previously mentioned Layout Components, this module has not been engineered up to attain real-time performances. However, to the best of our knowledge, our proposed system has achieved state-of-the-art performances with respect to similar on-line approaches.

Secondly, the idea on which this module is based is to introduce a first higher level semantic representation of the vehicle surroundings, allowing the Road Layout Estimation framework to gain situational awareness, one of the most crucial circumstances in urban scenarios [187]. The algorithm hinges on a stereo image stream, and it is our first step towards a complex outdoor urban scene understanding system by means of higher semantic level detectors. This work has been accepted to the IEEE International Conference on Robotics and Automation (ICRA) 2017.

4.1 The Intersection Model

As opposed to other state-of-the-art off-line methods, which usually require a batch processing of a short video sequence, our approach integrates the image data by means of an on-line procedure.

For the purpose of classification of the intersections, we consider the 7 common intersection configurations shown in Figure 4.2b as in the

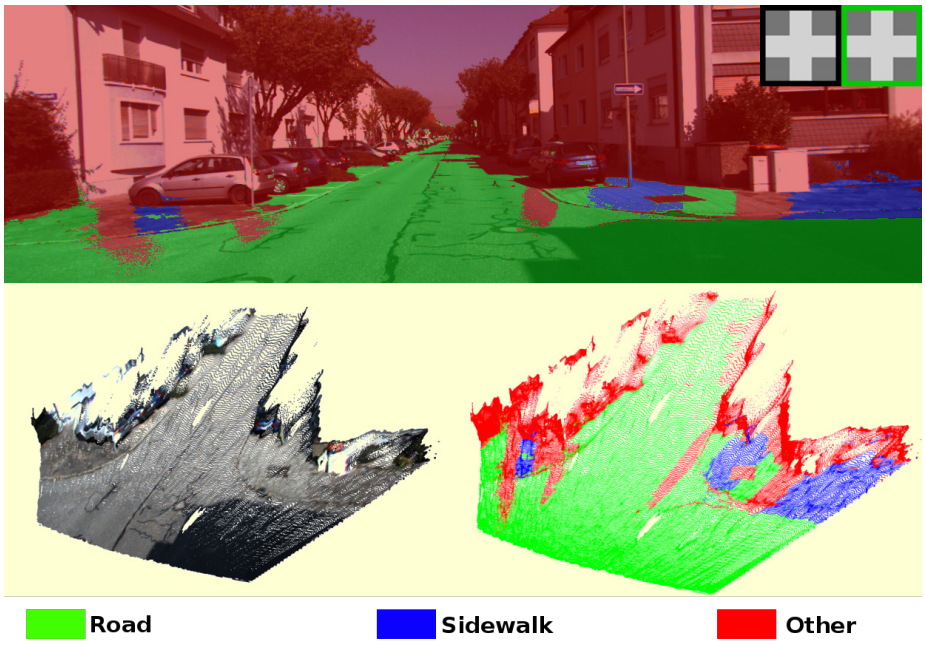
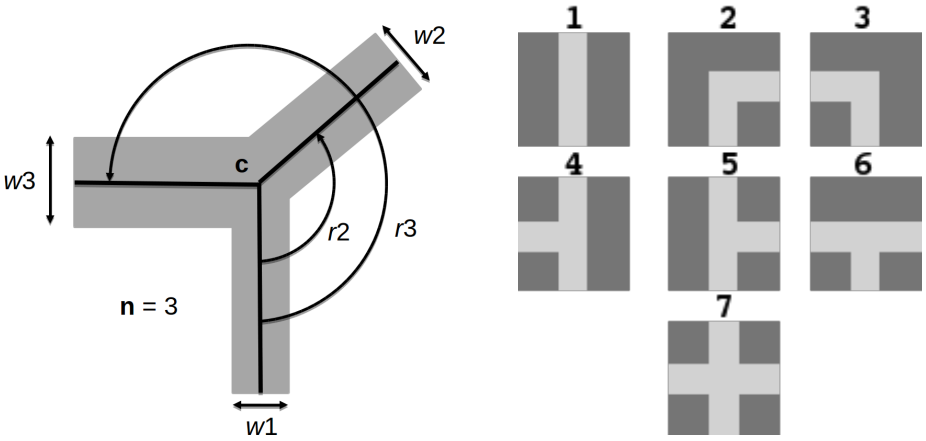


Figure 4.1: Example results from the proposed framework on one run of the KITTI dataset, and the corresponding road classification association. The image on the top represents the pixel-based classification. The two small boxes in the upper right corner respectively represent the true intersection topology and the one detected by our work. The bottom images represent the projection of the 3D geometric reconstruction (left) along with its 3D classification (right). Note that the intersection topology is correctly recognized even in presence of a few classification errors.



(a) (b)

Figure 4.2: In Figure 4.2a the geometry model used to generate the intersections and, in Figure 4.2b, the 7 common intersection patterns recognized by the proposed method (b).

state-of-the-art . In our proposed method a mixture of geometric and pixel-wise classification schemes has been employed to generate the intersection detection.

The results are used to score a hypothesized localization within the Road Layout Estimation framework, thus allowing to tackle the localization problem also by means of an analysis based on road intersection detection.

4.2 Geometric Segmentation of the Road

To achieve a good classification of the intersection topology, our method first need to identify the road surface. The pipeline starts computing the disparity map and the associated 3D point cloud by means of the ELAS [185] and the PCL [188] libraries. Differently from what proposed in Section 3.3, here the ELAS algorithm was preferred over the SGBM [181] algorithm since this method considerably reduces the noise amount with respect to the road surface as shown in Figure 4.3. The first step after the 3D point cloud reconstruction consists in the evaluation of the equation of the local road plane. For this purpose the resulting point cloud is filtered as follows:

- Crop to a bounding box, in order to keep only a small area in front of the vehicle. With respect to a x-axis forward, y-axis port and

z-axis up reference system, the bounding box limits were set using $\pm 5m$ as y lateral offset, $\pm 2.5m$ in z height coordinate, and x longitudinal range $0 - 20m$.

- Estimate the surface normal vectors for all the points within the bounding box area.
- Remove the points whose normal is not compatible with being a road. We consider suitable a normal vector if it is orthogonal with respect to the $x-y$ plane of the camera reference frame (in the used datasets the optical axis of the cameras are approximately parallel to the road plane). We accommodate reconstruction errors and inclined vehicle (with respect to the road and the suspension system) by including a tolerance threshold.
- Fit a 3D plane using the remaining points by means of a RANSAC approach and delete every point not close enough to the fitted plane equation.

4.2.1 Point Cloud Occupancy Grid: PCLOG

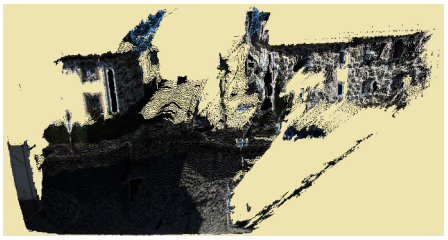
The whole point cloud is then exploited to generate the first component used for intersection detection: a discrete 2D bird-eye-view grid where each grid cell holds a probability value representing how likely is that the cell belongs to the road surface. We refer to this Occupancy Grid as *PCLOG*; an overview of the scheme is shown in Figure 4.4. The probability values of the cells are calculated by considering the average distance of the estimated 3D plane to the points falling inside each cell. The average is then weighed by considering a Probability Density Function of a standard Gaussian distribution with zero-mean and a variable standard deviation increasing within the range $0 - 50m$ (from $\sigma = 0.15$ at $0m$, *i.e.*, the position of the cameras, to $\sigma = 1$ at $50m$). Lastly, we normalize the values of all cells using Equation (4.1).

$$PCLOG[i,j] = \frac{PDF_d(\overline{x}[i,j])}{PDF_d(0)} \tag{4.1}$$

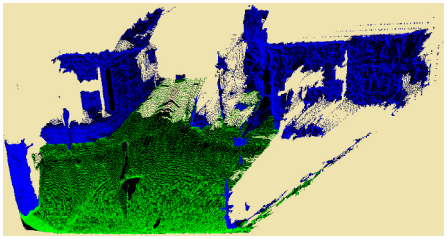
The output of this procedure does not allow us to reliably infer a stable intersection configuration, in spite of the fact that we did not observe critical errors from the fitting phase. The main reason can be traced to the slight height difference between the road surface and the sidewalks,

(a)

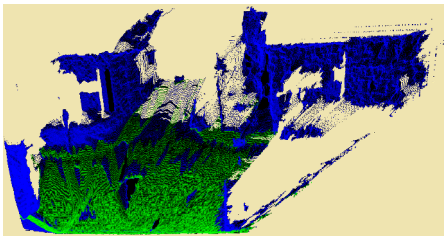
(b)



(c)



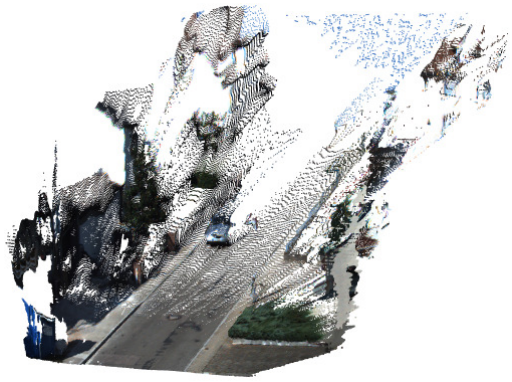
(d)



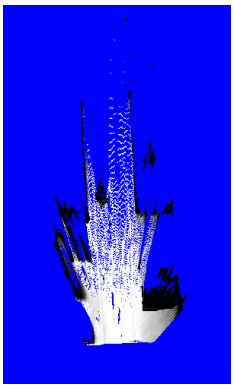
(e)

Figure 4.3: In this picture, a frame from the KITTI dataset and its PCL maps, obtained using the SGBM (4.3b) and the ELAS (4.3c) algorithms, respectively. The second approach yields significantly richer and denser road surface reconstructions even using the enhanced parametrization proposed in Section 3.3.3, allowing to achieve the best results from our geometric segmentation pipeline. In Figures 4.3d and 4.3e respectively, two classification results with the geometric reconstruction, using $0.15m$ and $0.05cm$ threshold distance from the 3D road plane equation.

(a) Original Left Camera Image



(b) Point Cloud from the ELAS algorithm



(c) The PCLOG

Figure 4.4

especially in the KITTI urban scenarios, this height is almost always very very small. A similar problem applies also to all the other objects laying on the same plane yet not belonging to the road or intersection area, *e.g.*, private roads, lawns, bicycle paths, etc. These situations, concurrently with the uneven pavement surfaces, leads to the treacherous situations presented in Figures 4.3d and 4.3e, due to choice of the threshold. Although this approach, based on the identification of a geometric road plane model, has shown interesting results in specific contexts, *e.g.*, roads with a low degree of clutter and well-defined curbs, an extended experimental activity clearly demonstrates that a reliable detector needs other information source. It has not come as a surprise, since state-of-the-art algorithms as [189] for road detection usually involves a mixture of 2D and 3D features, coupled with high level segmentation and classification techniques including Random Fields, Decision Trees or Neural Networks. However, as it makes no sense to renounce to benefit from this classification, we introduced a complementary image processing pipeline. In the following section we describe how image segmentation algorithms can aid the pure-geometric PCLOG approach.

4.3 Semantic Segmentation

To semantically segment the image we combine the per-pixel classifier proposed in TextonBoost [190] and the CRF based approach proposed in [69]. This combination involves a first per-pixel classification using the *texton* [191] strategy, followed by a second refinement stage by means of the CRF model. The aim of the second refining step is to introduce a spatial dependency between the pre-classified pixels, also known as pairwise potentials, and thus to better deal with scattered classifications, *i.e.*, a more precise boundary detections. This scheme was successfully used in other recent works [68, 192, 193, 194] and a comparison between the two classification steps is shown in Figure 4.5.

Following the approach introduced in [195], we applied a superpixel strategy for the CRF classifier rather than a per-pixel approach. The superpixel areas were evaluated using state-of-the-art algorithms including Quick-shift [196], SLIC [197] and the SLIC-zero variant, and the results are depicted in Figure 4.6.

Regarding the pairwise potentials interaction scheme, we leveraged the k-Extended Pairwise method using $k = \{2, 3, 5, 10\}$, weighing the

superpixels dependencies using contrast, center distance, disparity difference, and centroid misalignment. The results are discussed in Section 5.3. The training algorithm pseudo-code for the CRF is shown in Algorithm 4.

Algorithm 4 CRF Training

Require: Training-set images $\mathcal{D}$
Ensure: Trained CRF model
unary_on_sp($unary[i], super\_pixel[i]$) evaluates the average of the unary potentials of each pixel, within the same $super\_pixel[i]$
GT_on_sp($GT[i], super\_pixel[i]$) evaluates the unary class of the $super\_pixel[i]$ using the higher score
function LEARNCRF($\mathcal{D}$)
  $N \leftarrow$ SIZE_OF($\mathcal{D}$)
  $image[N] \leftarrow$ LOAD_IMAGE($\mathcal{D}$)
  $GT[N] \leftarrow$ LOAD_GT($\mathcal{D}$)
  $super\_pixel[N] \leftarrow \emptyset$
  $data\_train.X[N] \leftarrow \emptyset$
  $data\_train.Y[N] \leftarrow \emptyset$
  $superpixel\_type \leftarrow SLIC$" //or "quickshift" or "slic-zero"
  for $i \leftarrow 1$ to N **do**
    $super\_pixel[i] \leftarrow$ COMPUTE_SUPERPIXEL($image[i], superpixel\_type$)
    $unary[i] \leftarrow$ LOAD_UNARY($image[i]$)
    $data\_train.X[i] \leftarrow$ UNARY_ON_SP($unary[i], super\_pixel[i]$)
    $data\_train.Y[i] \leftarrow$ GT_ON_SP($GT[i], super\_pixel[i]$)
  end for
  $edge\_type \leftarrow$ "$pairwise$" //or "k-extended" or "fully-connected"
  ADD_EDGES($data\_train.X, edge\_type$)
  ADD_EDGE_FEATURES($data\_train.X, feature\_list$)
  $model \leftarrow EdgeFeatureGraphCRF$
  $model.fit(data\_train.X, data\_train.Y)$
  return $model$
end function

4.3.1 Training Dataset

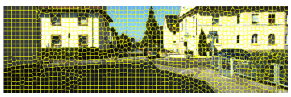
In order to train the proposed classification scheme, we exploit two publicly annotated dataset based on the KITTI dataset. First we used the dataset proposed by Sengupta *et al.* [68], which consists of 323 annotated images including the *road, sky* and *vertical* categories. The outcomes achieved after the training phase, however, were not satisfying as the road areas were frequently misclassified as sidewalks and viceversa. We also exploited the dataset related to the work of Alvarez *et*

(a) Original Image

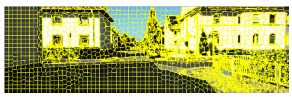
(b) Unary Classification

(c) CRF Classification

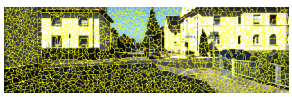
Figure 4.5: The figure depicts a semantic segmentation of the scene using the proposed method. In Figure 4.5a the original image from the KITTI dataset. In Figure 4.5b the resulting segmentation using the texton-based unary classification method. Finally Figure 4.5c depicts the integration of the pairwise potentials by mens of the CRF approach.



(a) SLIC-zero



(b) SLIC



(c) Quickshift

Figure 4.6: Comparison results using state-of-the-art algorithms for superpixel segmentation using same number of regions, in this example 1500.

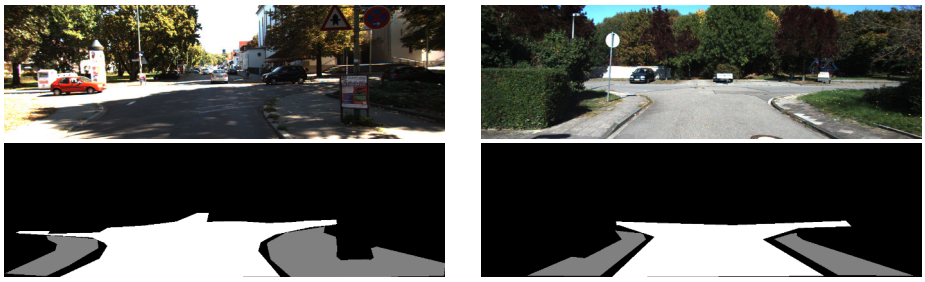


Figure 4.7: To better deal with intersection areas, we added a new set of images to the training set. The new dataset is available at: http://www.ira.disco.unimib.it/iralab/intersection-detector.

al. [198], by reducing the class space to include *road*, *sidewalks* and *other* only. Moreover, to better represent all the possible intersection types, we added 10 more images of intersection areas to the 70 available in the Alvarez dataset. An example of the set of images used to train the intersection detector is shown in Figure 4.7.

4.3.2 CRF Occupancy Grid: CRFOG

After the classification step, the pixels are projected in the 3D space following the same scheme used for the PCLOG. From these projection, we calculate a new occupancy grid that we call *CRFOG*. Each cell value represents the probability of the area to belong to a road and is computed as in Equation (4.2), *i.e.*, considering the ratio between the points classified as road with respect to the total number of point within each cell.

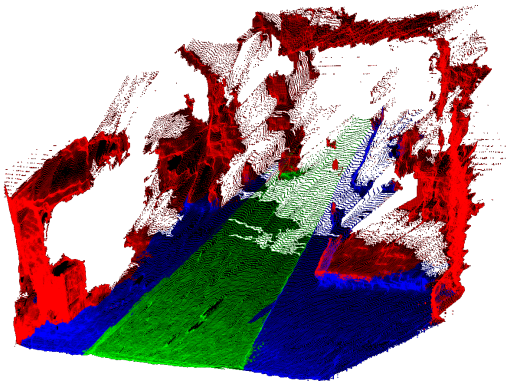
$$CRFOG[i,j] = \frac{n_{road}[i,j]}{N[i,j]} \tag{4.2}$$

Here $n_{road}[i,j]$ represents the number of points falling inside the $[i,j]$ cell of the occupancy grid that was classified as *road*. The value $N[i,j]$ represents the total number of points within the same cell.

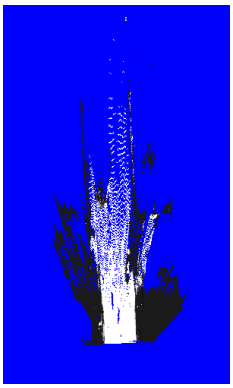
4.4 Temporal Integration

Perceiving the overall structure of an intersection from a single image is a hard task due to the unavoidable problem related to changes in the appearance. Moreover, processing every single frame with the aforementioned classifiers usually leads to unstable estimates, thus leading to

(a) Original Left Camera Image



(b) Reprojection of the Classified Image in the Point Cloud evaluated with the ELAS algorithm



(c) The CRFOG

Figure 4.8: The CRFOG creation pipeline. The resulting classification is overlayed to the original left-camera image in Figure 4.8a. The Figure 4.8b shows the resulting classified point cloud where the green color represents the road, the blue the sidewalks and with red the vertical category.

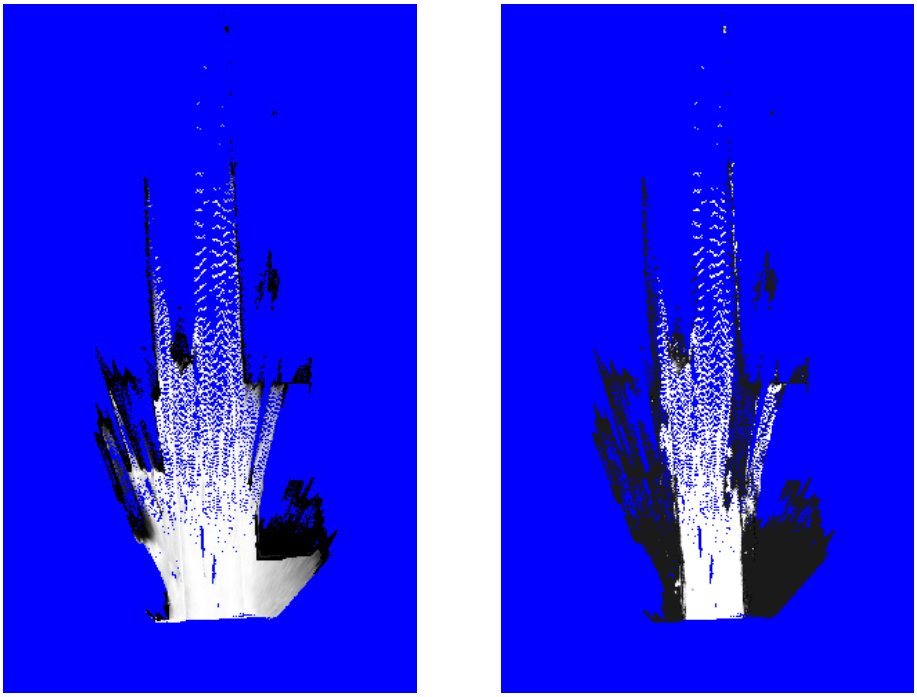
temporary unstable occupancy grids. We tackle this problem by leveraging the temporal coherence between road models computed in consecutive frames, and corroborating the results in both our occupancy grids. To handle probabilities derived from multiple knowledge we evaluated the following methods for a more detailed description of the three integration approaches.

- Bayesian derived approach
- Evidence Theory, *i.e.*, Dempster–Shafer Theory
- Proportional Conflict Redistribution rule no. 6 (PCR6) derived from the Dezert-Smarandache Theory D(SmT) [199].

Although we refer the reader to Section 5.3.3 for a numerical comparison of the results, Figure 4.9 shows the best outcomes, achieved exploiting the PCR6 rule. From a technical perspective, the approach relies on a set of temporally integrated occupancy grids that are updated after every detection by means of the PCR6 rule. According to the Dempster-Shafer theory, each cell of the occupancy grid contains the p probability value of being *road area*, the q value ($q = 1 - p - u$), *i.e.*, the probability of the cell being *not-road*, and the binary value u representing whether or not the area is *unknown*, see *e.g.*, [200]. These considerations allowed us to correctly handle the unknown space, which cannot be handled using a simply Bayesian approach.

4.5 Increasing the classification consistency

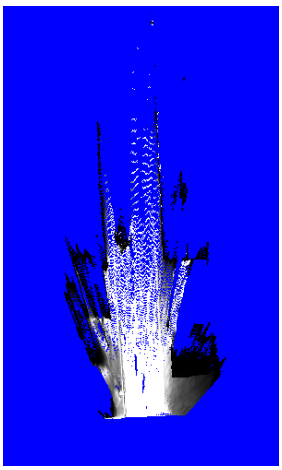
In order to further increase the temporal coherence of the classification estimates in both the PCLOG and the CRFOG, a temporal hysteresis on the classification values has been introduced, so that multiple same-class classifications increase the classification belief over each cell of the occupancy grids. We integrate a dual-counter scheme as follows. For each cell, the first counter sums how many times the area has been observed (for many reasons, *e.g.*, occlusions, the 3D reconstruction might not be able to observe points in some area). The other counter starts from the allowed number of consecutive frames the area will be kept as valid even if not observed, and it is decreased each time the area is not observed. Finally, we reset it when it gets a new observation. If this counter reaches zero the classification of the cell is reset to unknown. The allowed number of consecutive no-observation frames is bounded on both sides; in



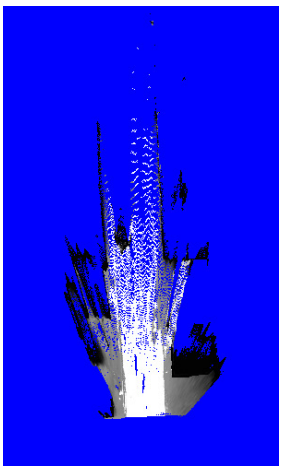
(a) PCLOG

(b) CRFOG

(c) Bayes Integration



(d) Dempster-Shafer Integration



(e) PCR6-Rule Integration

Figure 4.9: In the picture, an integration example between the PCLOG Figure 4.9a and the CRFOG Figure 4.9b using respectively the a Bayesian approach, the Dempster-Shafer theory and the PCR6 rule. The blue areas represent the unknown space (only in Dempster-Shafer and PCR6).

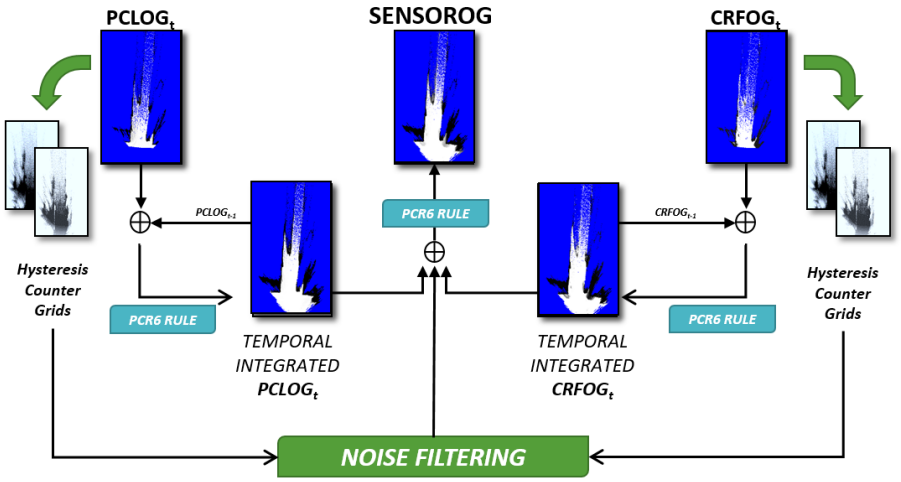


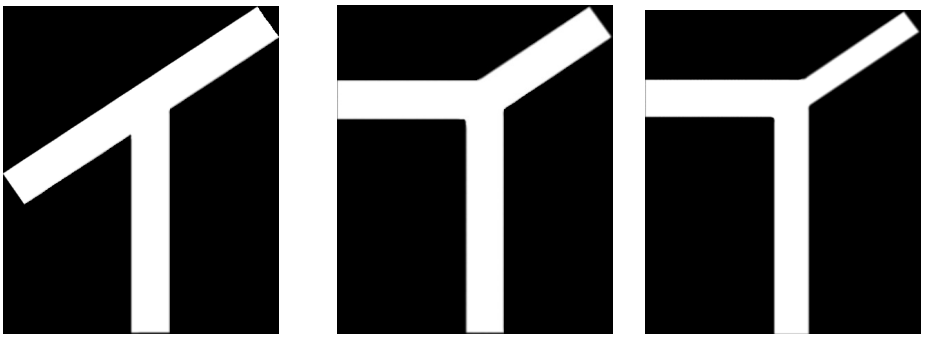
Figure 4.10: The SENSOROG is computed using the results from the PCLOG and the CRFOG, expoiting the PCR6 rule, after a further noise-filtering procedure performed on the both occupancy grids.

between it equals the current value of the first counter. The two occupancy grids are then joined using again the PCR6 rule, resulting in a new occupancy grid called *SENSOROG*. The overall scheme used to evaluate the final SENSOROG is depicted in Figure 4.10.

4.6 Scoring Function and Classification

To evaluate a hypothesized upcoming intersection using the obtained sensor measure, *i.e.*, the SENSOROG, we leverage the proposed intersection model shown in Figure 4.2a. We took our inspiration from the work presented by Geiger [20], by extending it to accommodate not symmetric intersections like in Figure 4.11. Moreover, we increased its expressiveness by allowing the model to represent intersections with more than 4 incidence roads and different widths for each road. The model has the following parameters, which are dynamically determined with respect to the vehicle position:

- The distance **c** of the intersection center, with respect to the vehicle position;
- The number **n** of road segments (arms) involved in the intersection;
- For each arm i, the width w_i and its orientation r_i with respect to the current road segment.



(a) Original Geiger Proposal. Please notice the symmetry in the front road.

(b) Our proposal, without symmetry in the front road.

(c) Our proposal, with different road widths.

Figure 4.11: The figure depicts different intersection configurations represented using our model. A comparison between the model proposed by Geiger [20] is presented in Figures 4.11b and 4.11c with respect to the original proposalFigure 4.11a, where symmetry and width are evaluated using a *per-road* scheme.

Using this model we are able to generate almost every type of intersection, except the ones that can be better represented by roundabouts. The model allows us to generate a new set of occupancy grids, called *EXPECTEDOG*s, which represents a set of hypothesized intersections. These new grids, which are homogeneous to the SENSOROG, allows us to tackle the intersection geometry classification using a probabilistic basis, thus modeling the real-world uncertainties. The EXPECTEDOG represents an expected *sensor* reading, and the comparison of the two enables us to accommodate the uncertainties of our detector. From a technical perspective, as the two grids are represented as images, we can score every EXPECTEDOG by means of a similarity check with respect to the SENSOROG grid. We exploited the *normalized correlation coefficient* function, which have been proved to be give good results even for complex scenarios.

This comparison allows us to make two considerations. On the one hand, measuring the likelihood of the SENSOROG with respect to a hypothesized position of the vehicle allow us to discriminate different vehicle positions by means of a new Layout Component. Let us consider a very cluttered scenario where the previous Layout Components fail to disambiguate two vehicle positions as it might happen within a city-like scenario. In such situation, detecting the configuration and the distance from the center of the intersection would help us.

On the other hand, we can also exploit the proposed detection pipeline to evaluate the performances of the approach with respect to the state-of-the-art algorithms in the intersection recognition field. These algorithms usually clusterize the feasible configurations using the 7 patterns shown in Figure 4.2b. Our results are presented in Section 5.3.

4.7 Conclusions

In this chapter, we have presented an probabilistic approach for the detection and the classification of upcoming road intersections. As a first step towards a semantic analysis of the scene, the algorithm aims at classifying the road topology by means of dual classification, geometry and pixel-level. The main contributions are:

- A pure geometric evaluation of the road surface through a 3D point-based scene reconstruction pipeline. The pipeline leverages the ELAS algorithm and a point cloud processing phase, but fails to effectively discriminate sidewalks areas and generally all roads without road boundaries clearly marked by height discontinuities. A Point Cloud Occupancy Grid is generated in this phase (PCLOG).

- We then introduce a second classification step within a Conditional Random Field (CRF) approach, where an image analysis process is performed. We exploit the Texton approach for the unary potentials and a superpixel scheme for the pairwise potentials of the CRF, generating a second classification that we called CRFOF.

- The calculated occupancy grids are integrated over time using the PCR6 rule in order to handle the uncertainties and the missing information. The integrated grids are then fused into the final SENSOROG.

- Using the vehicle position gathered from the Layout Component and the OpenStreetMap data, we generate the EXPECTEDOG using an enhanced intersection model that we derived from [20]. The EXPECTEDOG is then matched with respect to the SENSOROG, allowing us to discriminate the intersection type with respect to the 7 common patterns found in the literature. Moreover, we formalized a new Layout Component (road intersection) for the Road Layout Estimation framework, which that potentially enables the

framework to disambiguate localization hypotheses by means of the analysis of the upcoming intersection.

Chapter 5

Experimental Evaluation

In this chapter, we present the experimental evaluation of the Road Layout Estimation framework and the detections of Layout Components presented in the previous chapters.

In Section 5.1 we present a first evaluation of the proposed framework. To test the effectiveness, we implemented a map-matching Layout Component that allow the system to localize a vehicle in urban road settings, using only the road graph retrieved from the OpenStreetMap service and the Visual Odometry from the LIBViso2 library.

In Section 5.2 we experiment with our second proposal, which leverages the building's outline information stored within the OpenStreetMap service. Using this new detector and the associated Layout Component we show how to decrease the vehicle position in both the lateral and the longitudinal uncertainty , achieving a lane-level accuracy in positioning in urban areas.

In Section 5.3 we experiment with our intersection detection pipeline. A comparison with other state-of-the-art approaches shows that our approach achieved a significant gain in terms of detection accuracy.

In order to evaluate the proposed framework, we used the challenging KITTI dataset [201], whenever possible. Regarding the intersection detector, we introduced a new set of annotations that are publicly distributed[1] for further comparisons and evaluations in the context of intersection detection.

In Section 5.4 we experiment with the lanes and width tags of OpenStreetMap, by evaluating the results of a simple line detector and tracker.

[1] http://ira.disco.unimib.it/iralab/intersection-detector

All the sections provide a critical interpretation of the achieved results, suggesting the advantages of the approaches yet stressing the main weak points.

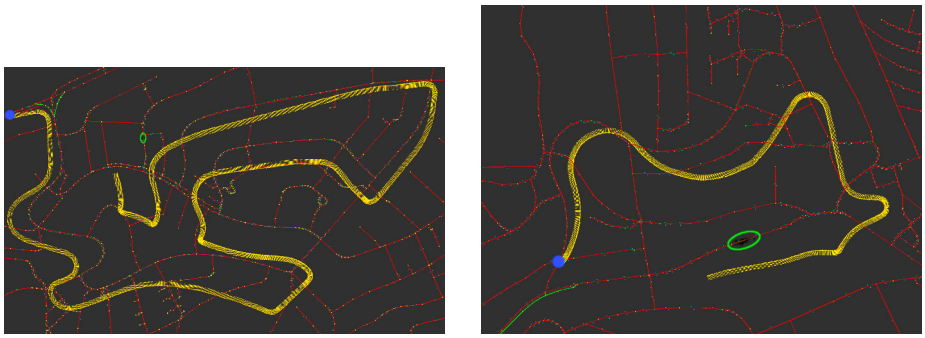
5.1 OpenStreetMap Matching Pipeline

In Section 3.2 we demonstrated the flexibility of the Road Layout Estimation framework by applying it to the problem of vehicle localization. Here we prove the effectiveness of our proposal by comparing the localization accuracy achieved by our framework in comparison with the similar, yet state-of-the-art method presented in [9]. We tested our approach on ten well-known KITTI [201] sequences chosen from the roads and residential categories with complex road scenarios, in order to stress the system with non-trivial environments. To evaluate the localization capabilities, for each dataset sequence we initialized the localization hypotheses spreading them with a normal bivariate distribution, centered on the first available GPS position. We used a purposely amplified uncertainty of 60 meters as 3 σ, in order to scatter the initial hypotheses on a wide area. Then, a *lock-on-road* procedure was performed, moving and aligning position and orientation of the Layout Hypotheses to the nearest road segments. In the case of a two-ways road, a pair of Layout Hypotheses were then generated, one for each driving direction, in order to cope with the GPS lack of orientation (during initialization we are using one GPS measure only). During the experiments, we used a constant number of 80 Layout Hypotheses and ran all the datasets at natural speed. Using this configuration, we achieved nearly real-time performances, with a processing frequency of about 9.6 Hz, very close to the KITTI stereo camera frequency of 10 Hz. Figure 5.4 shows an overview of the localization results on some tested KITTI dataset sequences, while Table 5.1 summarizes the localization accuracy results in terms of the root mean square error (RMSE). In particular, the last two columns represent the calculated RMSE between the vehicle pose estimated by our framework and, respectively, the OpenStreetMap road graph information and the GPS-RTK groundtruth.

Beside the comparison with respect to groundtruth, we also present the OpenStreetMap road graph comparison. The reason, as further discussed in Section 5.1.2, is that OpenStreetMap road graphs are designed to represent the geo-localized center of the road. Therefore, since our

Table 5.1: Proposed framework comparison with respect to the ground truth

Sequence Reference Name	Sequence Lenght	Category	GPS-RTK (m) track length	Final position error (m)	Final position percentage error	RLE RMSE (m) wrt OpenStreetMap	RLE RMSE (m) wrt GPS-RTK
2011_10_03_drive_0027	7:50	Residential	2,651.92	1.34	0.04%	0.92	0.73
2011_10_03_drive_0042	1:54	Road	2,448.34	18.30	0.75%	1.11	1.88
2011_10_03_drive_0034	8:03	Residential	2,872.90	5.40	0.19%	1.00	1.50
2011_09_30_drive_0016	0:28	Road	385.76	10.16	2.65%	0.79	3.48
2011_09_30_drive_0020	1:53	Residential	1,227.57	2.38	0.11%	0.89	1.40
2011_09_30_drive_0018	4:47	Residential	2,205.77	1.59	0.13%	0.92	1.50
2011_09_30_drive_0027	1:53	Residential	693.12	3.94	0.57%	0.89	0.92
2011_09_30_drive_0028	7:02	Residential	3,204.46	24.55	0.76%	0.91	1.93
2011_09_30_drive_0033	2:44	Residential	1,700.71	11.71	0.69%	0.97	1.66
2011_09_30_drive_0034	2:04	Residential	918.99	3.43	0.37%	0.92	1.84
TOTAL	38:38		19,374.68				
AVERAGE	3:51		1,937.46			0.93	1.68
TOTAL in [9]	28:32		7,172.83				
AVERAGE in [9]	2:02		512.34				7.39



(a) (b)

Figure 5.1: Here the accumulated Visual Odometry error is shown. We used the well known LIBViso2 [202] library. The blue area identifies the sequence dataset starting point, the yellow trace the LIBViso2 estimated vehicle position and the green ellipse the estimated final position area (3σ). Please note that here the trace represent the x-axis arrows in an z-forward reference system and thus it is orthogonal to the vehicle forward motion.

system uses this information to compensate the Visual Odometry error cumulation (as depicted in Figure 5.1), the resulting localization estimate has a systematically offset of half the size of the roadway as the KITTI vehicle was moving inside a lane and not on the centerline. This is summarized by the two average values over the ten dataset sequences, *i.e.*, 0.936 m with respect to the OpenStreetMap topometric map and 1.688 m with respect to the GPS groundtruth. When compared to the results of [9] (7.396 m), the localization accuracy of our framework is more than four times more precise relating to the GPS groundtruth and an order of magnitude with respect to the OpenStreetMap reference.

5.1.1 Weak Points

The main weakness of the algorithm is related to the systematic offset introduced by the OpenStreetMap service. Since our system uses the road graph as a reference in the scoring function, *i.e.*, the likelihood of a hypothesis close to a road segment is higher, a misalignment in the road graph leads to a systematic bias. Common scenarios include decentralized road segments, abrupt curves or intersections, where the OpenStreetMap road graph does not approximate the smooth curvature of the road. An example of the curve situation is detailed in Figure 5.2. As a drawback, we had a slight inflated error in proximity to intersection areas, where the pure geometric road graph does not approximate

the smooth car trajectories followed by the driver. Despite this artificial injected error, promising results were obtained by the framework. Nevertheless, even considering that these misleading conditions could be substantially reduced by the forthcoming high definition maps, we argue that a robust localization algorithm needs to take into account also other road features, as to reduce the aforementioned systematic error and thus to achieve a more accurate lane-level localization.

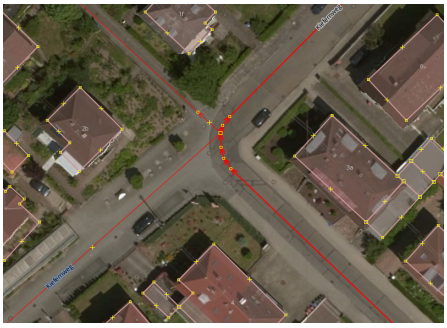
It is important to notice that in case of complex road network scenarios, even with high injected uncertainties the framework successfully localized the vehicle after a modest distance covered, implying that the framework could also cope with a rough global localization task even without any further Layout Component, getting closer to the results obtained by more complex vision-based approaches available in the literature like [105].

5.1.2 Discussion

Relying on topometric maps as the only global information to perform road vehicle localization may lead to indiscernible situations. For example, let us consider the situation depicted in Figure 5.3. The red line represents the groundtruth path the vehicle actually traveled, while the yellow lines are the road segments from the topometric map. As it can be seen, the vehicle has been traveling in the rightmost lane for a while, eventually turning onto the ramp, while the map road segment of the ramp splits just immediately before the ramp. Evaluating the localization hypotheses with respect to the road segments unavoidably leads to an inaccurate localization estimate, represented by the green line. This kind of treacherous situations arise very frequently even on ordinary, non-splitting roads, where the map road segments run in the very center of the roadway, while vehicles are of course driving in lanes. In other words, while we have normally a match at the symbolic level between map and sensor data, we might have from time to time a semantic gap. Considering the case above, we have a semantic gap in the lane number as the vehicle travels the part of the road with an extra lane, which will become the ramp. The modularity of the proposed framework provides a means to tackle such disambiguation problems. For the case above, leveraging a state-of-the-art line detector, would allow to cope with the displacement between the topometric map (roadway center) and the actual lane the vehicle is driving in. An alternative solution may take into

(a)

(b)



(c)

Figure 5.2: Figure 5.2a: small localization failure introduced by using the OpenStreetMap information to compensate the Visual Odometry error cumulation. The red line represents the groundtruth path of the vehicle, while the localization estimate is represented by the green line. The error is due to the coarse road model in OpenStreetMap (see Figure 5.2b), as opposed to the smooth trajectories followed by the vehicles. Figure 5.2c shows a flawless yet uncommon road approximation.

account the width of the road. These two considerations will be discussed in Section 5.4.

5.2 Buildings Localization Enhancements

This section discusses the results achieved using the Building Detector and the associated Layout Component presented in Section 3.3. The algorithm aims to enhance the initial rough road-level localization shown in the previous section, by means of a pure geometric building façades detector. The results are matched to the building outlines provided by the OpenStreetMap service, in order to reduce the uncertainty of the localization.

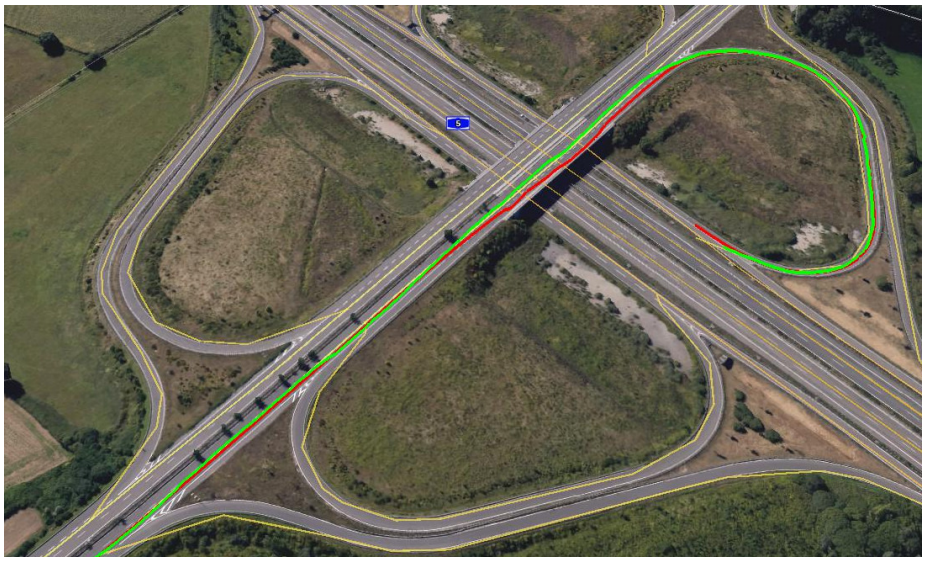
We tested our approach on 11 sequences from the well-known KITTI benchmark suite, choosing *urban* and *city* sequences containing both basic and challenging scenarios. During the evaluation activity, we initially downloaded all the buildings outlines of the involved area from the OpenStreetMap service, in order to avoid any network delay. For each dataset sequence, we initialized the localization hypotheses spreading them with a normal bi-variate distribution, centered on the first available GPS position, using a slightly amplified uncertainty of 2 meters as 3σ, in order to scatter the initial hypotheses on a road area. Differently from the tests in Section 5.1, here we aim to stress the lane-level localization performances and thus higher uncertainties are not required. We would have initialized as before, and then excluding from reporting the transient of the previous section localization system, with the only net result of loosing part of the evaluation dataset.

5.2.1 Experimental results

We started the experimental activity exploiting the previous OpenStreetMap Matching Layout Component, which allows us to have a rough, yet *locked-on-road*, localization estimate by means of a road-network localization. The error of this system can be considered to have reached its regime, because of the more accurate localization. Then, to evaluate the localization enhancement, we activate the Building Layout Component letting the framework combine the building information with the road graph.

Figure 5.5 shows an overview of the localization improvements on some tested KITTI dataset sequences, while the Figure 5.6 depicts how

(a)



(b)

Figure 5.3: Figure 5.3a depicts an example of a treacherous situation where the topometric information provided by OpenStreetMap is not sufficient to correctly localize the vehicle. The red line represents the groundtruth path the vehicle, while the yellow lines are the road segments from the topometric map. In such a situation, evaluating the localization hypotheses with respect to the road segments unavoidably leads to the inaccurate localization estimate represented by the green line. Figure 5.3b shows how the localization estimate fail to correctly track the real position of the vehicle once the vehicle turns to the exit lane.

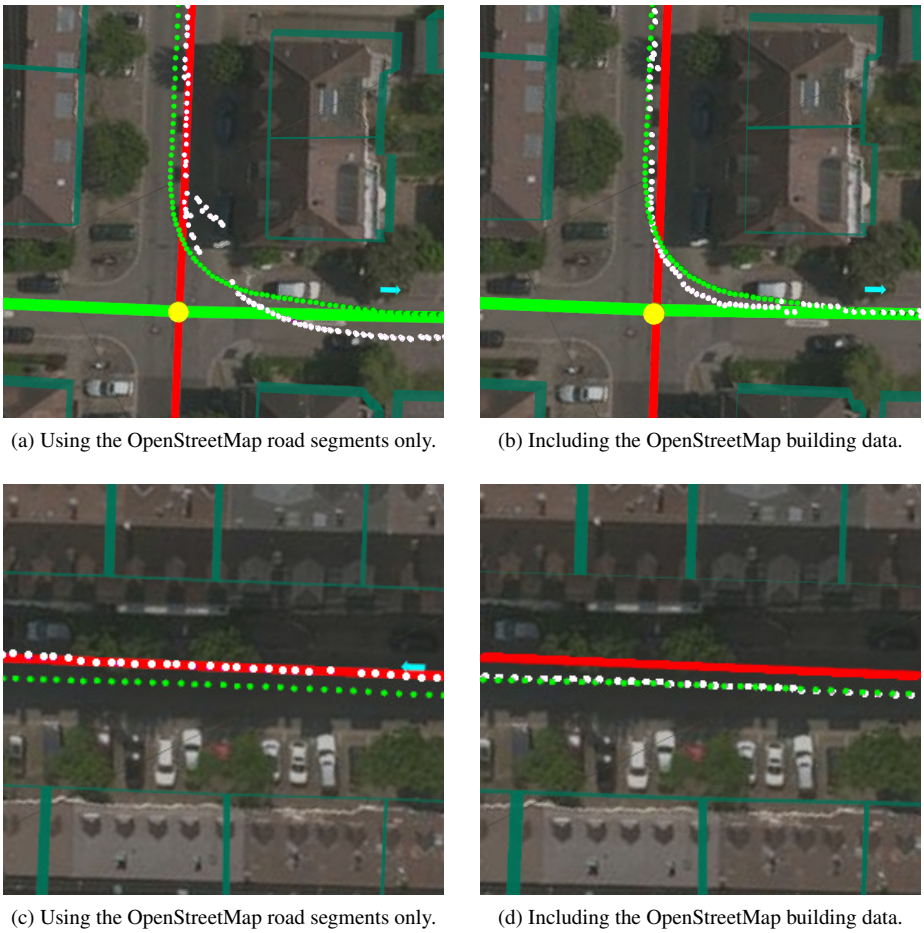
Figure 5.4: Localization results on the KITTI dataset. The red line represents the groundtruth path of the vehicle, while the localization estimate is represented by the green line.

Table 5.2: Kitti sequences used for experimental evaluation of the Buildings Detector

Sequence Name	Category	Sequence Length (mm:ss)	Sequence Length (m)
2011_10_03_drive_0027	Residential	7:50	2651.92
2011_10_03_drive_0034	Residential	8:03	2872.90
2011_09_30_drive_0018	Residential	4:47	2205.77
2011_09_30_drive_0020	Residential	1:53	1227.57
2011_09_30_drive_0027	Residential	1:53	693.12
2011_09_30_drive_0028	Residential	7:02	3204.46
2011_09_30_drive_0033	Residential	2:44	1700.71
2011_09_30_drive_0034	Residential	2:04	918.99
2011_09_26_drive_0005	Residential	0:16	66.10
2011_09_26_drive_0046	Residential	0:13	46.38
2011_09_26_drive_0095	Residential	0:27	252.63
TOTAL		36:16 mm:ss	15475.44 m

Sequence Name	RMSE (m) Road Graph Only	RMSE (m) Road Graph and Building Data
2011_09_26_drive_0005	2.52	1.92
2011_09_26_drive_0046	2.40	1.64
2011_09_26_drive_0095	2.66	1.47
AVERAGE	2.53 m	1.68 m

Although we evaluated the building detection pipeline on all the sequences shown in the first part of the table, here we report only the results in the last three sequences in which our detection manage to increase the localization accuracy. The strong presence of vegetation, as well as other cluttering elements (as depicted in Figures 5.7a to 5.7c), in combination with our pure-geometric approach, have shown sub-optimal results that lead us to believe that a reliable building detector has to consider also typical façades features by means of image processing, thus enhancing the geometric plane detection pipeline proposed in this work. We nevertheless obtained remarkable results in good scenarios like Figures 5.7d and 5.7e, and the system was able to cope with ambiguous scenarios, and localize the vehicle accurately.



(a) Using the OpenStreetMap road segments only.

(b) Including the OpenStreetMap building data.

(c) Using the OpenStreetMap road segments only.

(d) Including the OpenStreetMap building data.

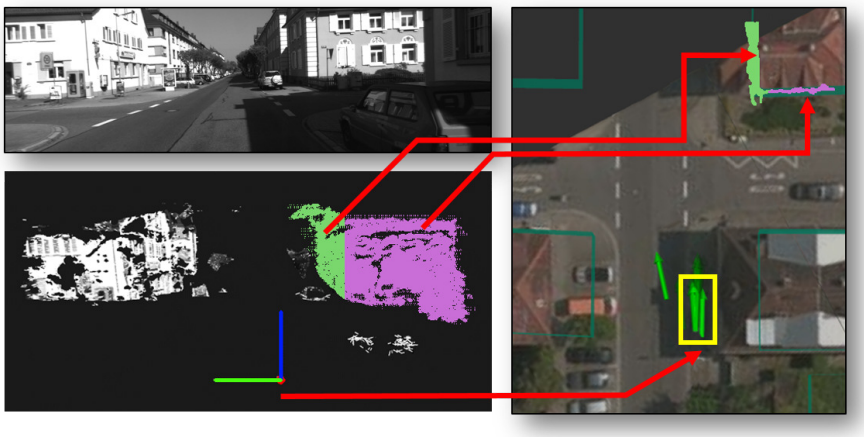
Figure 5.5: The results of a localization run. The red and green straight lines represent the OpenStreetMap road segments. The green dots represent the groundtruth path of the vehicle, while our localization estimate is represented using white dots. In Figure 5.5a performance using only the road segments, our previous approach. The misaligned dots are due to the coarse road model in OpenStreetMap, as opposed to the smooth trajectories followed by the vehicles. Figure 5.5b depicts the results of the proposed approach. The misalignment reduction is clearly visible in Figures 5.5c and 5.5d. Please note that the systematic offset with respect to the satellite image is due to an approximation of the latitude and longitude of our visualization software.

the Layout Component allow the framework to score a localization hypothesis in the proximity of an intersection area, thus exploiting not only the lateral buildings but also the structures on the opposite side of the crossing.

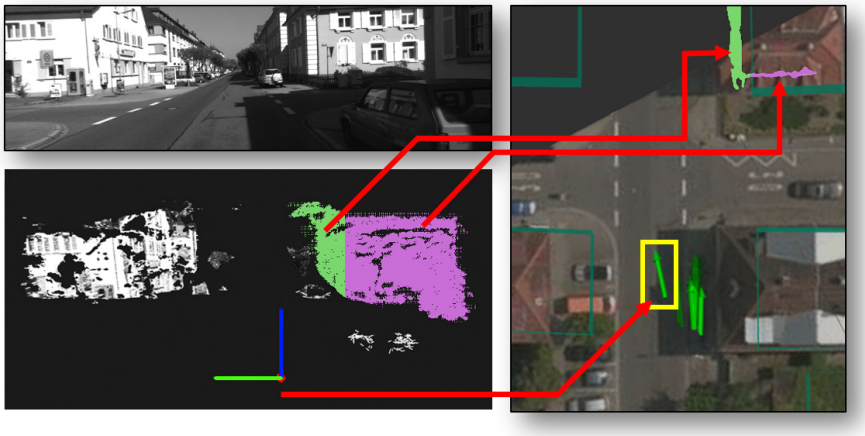
In Table 5.2 we present all the sequences we used in the experimental activity, which includes residential areas from the Karlsruhe city in Germany. The second part of the same table summarizes the localization accuracy results in terms of the Root Mean Square Error (RMSE) with respect to the best results achieved within our test. In particular, the last two columns respectively represent the calculated RMSE between the vehicle pose estimated by the framework without the building component and the same framework with the building component activated. Beside the comparison with respect to ground-truth, Figures 5.5c and 5.5d shows how the building detections can reduce the lateral uncertainty while the vehicle is traveling on the road so as to achieve a remarkable lane-level localization. Moreover, since crossing areas are usually surrounded by buildings, in the context of residential areas, the algorithm also provides an enhancement in terms of longitudinal localization by smoothing the trajectories arising in the no-building algorithm, *i.e.*, leveraging the OpenStreetMap network only.

5.2.2 Discussion

The proposed algorithm presents the following drawbacks. First, the façade detection relies on a purely geometric pipeline, which is based on the point-cloud calculated from the stereo images. Since we did not introduce any clutter removal algorithm, it follows that misleading scenario configurations may lead to wrong plane detections, as depicted in Figures 5.8a and 5.8b. Other potential errors arise from incorrect perceptions like in Figure 5.7, where planes are fitted over hedges and dense fences. A second and more specific issue arises from the geometric approach. Since we compare plane-to-plane distances, *i.e.*, modeling the façade plane equations from the point cloud and, on the other hand, generating them from OpenStreetMap outlines, a misalignment between consecutive buildings such as in Figure 5.8c may introduce a bias in the localization process and thus lead to indiscernible hypotheses, even considering the distance from the road center. The problem is related with the infinite planes equations and the *winner-takes-all* strategy described in Section 3.3.6. An example of this problem is depicted in Figure 5.8c



(a) In this figure, the building façades are projected in the 3D space considering a good localization hypothesis and thus leading to a good alignment with respect to the OpenStreetMap outlines.



(b) In this figure the 3D projections are misaligned due to the incorrect localization hypothesis.

Figure 5.6: In the figures we show the matching process that allows the algorithm to score the localization hypotheses. In both the figures, the yellow box on the right part of the image identifies the Layout Hypothesis which is being evaluated. As can be easily noticed, in Figure 5.6a the detection of the two façades is almost overlapped with respect the the OpenStreetMap outlines while in Figure 5.6a a slight misalignment may be observed. Please notice that both the figures show the points associated to the façade rather than the plane equation.

where the big green and blue lines represent the infinite edges of the façades gathered from the OpenStreetMap, which are associated with the buildings with labels A and B. Under these circumstances the localization hypothesis H_1 represented with the red car has a lower score with respect to the hypothesis H_2, since the likelihood of the distance measured by the detector is greater in $H2$. On the one hand, a possible solution to this issue may arise from a smarter comparison which takes into consideration also the front distance of the building façade. On the other hand, we may tackle the issue introducing a probabilistic approach, by means of a sensor model aimed at handling this kind of uncertainties and thus allowing the framework to handle these treacherous situations. In spite of the aforementioned workarounds, we believe that reliable solution would also necessarily include an additional image processing step aimed at disambiguate also the problem of Figures 5.8a and 5.8b.

Considering on the weak side, we note that the detection pipeline has not yet achieved real-time performances. One of the main issues is lies in the façades research phase, in which the Region Growing Segmentation algorithm drastically limits the performance of the detection, consuming nearly the 70% of the time, as depicted in Figure 5.9.

Related to geometric pipeline and its parametrization, in this *per-frame* analysis we do not include any temporal clue, *e.g.*, a temporal integration of the perceived 3D point-cloud. Despite these limitations, the algorithm has proved its effectiveness in the context of urban loosely-cluttered environments, enhancing the localization accuracy and compensating the rough localization achievable using only the road network as proposed in Section 5.1. Therefore, although we would classify this part of the work as in progress, we believe we demonstrate this to be an effective visual clue, in order to achieve lane-level localization accuracy.

(a)

(b)

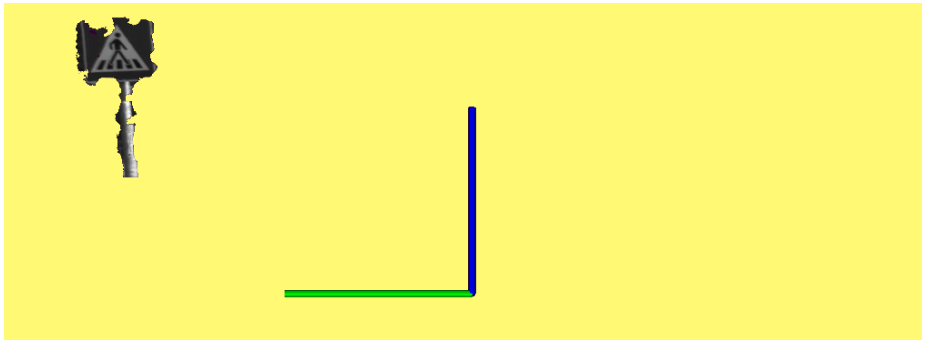
(c)

(d)

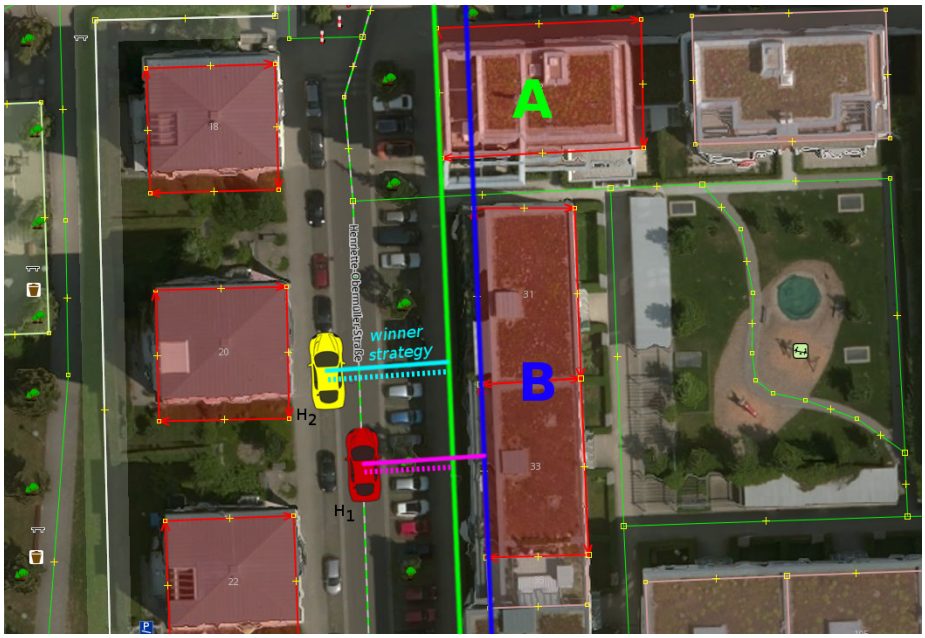
(e)

Figure 5.7: In Figures 5.7a to 5.7c typical scenarios where the pure geometric pipeline fails to fit geometric façade plane models. The issues arise from incorrect perceptions on hedges and fences. Figures 5.7d and 5.7e depicts two positive contexts, in which the proposed algorithm modeled properly the building's façades.

(a)



(b)



(c)

Figure 5.8: The figure depicts an erroneous classification with a pedestrian sign misinterpreted as a façade. Due to the thresholds used in the segmentation phase and the pure-geometric pipeline, the buildings on the right contains less 3D points than the ones belonging to the sign, causing the procedure to reject the point lying on the façade. Figure 5.8c depicts a toy example where the plane-to-plane problem is highlighted.

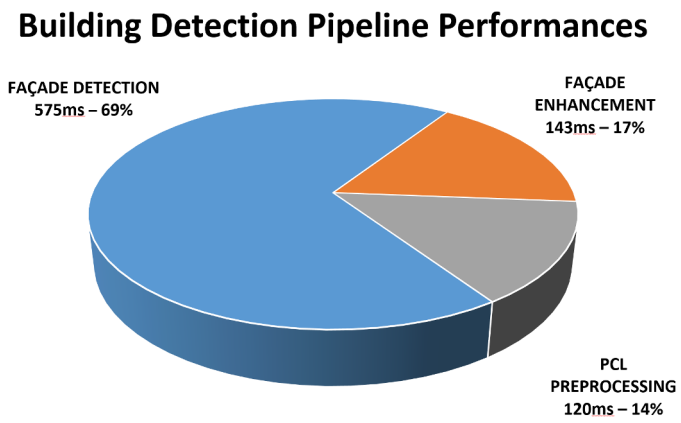


Figure 5.9: The figure depicts the most time-consuming phases of the Building Detection Pipeline. We recall that the pre-processing phase consists of two phases (cropping and surface normal evaluation) and the façade enhancement is aimed to extract building clusters from the surfaces within the Façade Detection phase (which involves the Region Growing Segmentation algorithm).

5.3 Intersection Detection

This section presents the results of our on-line intersection topology detector introduced in Chapter 4. The processing pipeline in this case was tested with an exhaustive experimental activity aimed at proving its validity in urban scenarios. We validated our system on the 8 sequences shown in Table 5.3, taken from the KITTI dataset [201]. Unlike the other detectors presented in this thesis, which were integrated within the Road Layout Framework as Layout Components, here we focus on achieving the best intersection topology classification. The reason is related to the insufficient results of our initial intersection detection pipeline, which leveraged a pure geometric scheme similar to the building detection pipeline presented in Section 3.3. Accordingly to this consideration, the experimental activity related to section of is not focused on timing performances. To evaluate the classification performances, we chose challenging residential sequences including different road configuration scenarios, as to demonstrate the detector classification capabilities with respect to the 7 crossing patterns shown in Figure 5.10. The ground truth used for the assessment was created by manually annotating each frame of the sequences with the appropriate topology. In particular, we set the topology label [1..7] when the vehicle is approaching the intersection, *i.e.*, when it is less than 30 meters from the intersection. Moreover, we

Figure 5.10: The 7 crossing topologies usually used in state-of-the-art approaches.

Table 5.3: Kitti sequences used for experimental evaluation of the intersection detector

Sequence Reference Name	Sequence	Category	GPS-RTK (m)
2011_10_03_drive_0027	7:50	Residential	2,651.92
2011_10_03_drive_0034	8:03	Residential	2,872.90
2011_09_30_drive_0018	4:47	Residential	2,205.77
2011_09_30_drive_0020	1:53	Residential	1,227.57
2011_09_30_drive_0027	1:53	Residential	693.12
2011_09_30_drive_0028	7:02	Residential	3,204.46
2011_09_30_drive_0033	2:44	Residential	1,700.71
2011_09_30_drive_0034	2:04	Residential	918.99
TOTAL	36:16		15,475.44

included in the classification a flag (called *crossing flag*) that indicates whether the vehicle is moving within the intersection boundaries. We use the flag to make it clear when the topology "is hidden" by the vehicle being into the intersection.

Believing that further research is required in the road intersection detection context and to allow future researchers to compare their work with respect to ours, we also published our KITTI annotations on-line[2] and some of these annotations are shown in Figure 5.11.

5.3.1 Experimental Results

With respect to the experimental setup, we made the following considerations. During the evaluation activity, as to achieve comparable results with respect to the state-of-the-art method proposed by Ess *et al.* [21], we limit the ground truth to a distance with respect to the intersection no less than 20 m. It follows that the results of the detector are compared only in straight roads and up to 20 meters from the crossing area. Moreover, since at this time the detections are not meant to enhance a localization hypothesis, *i.e.*, the system is not yet integrated as a Layout Component, we evaluated the detector using the ground truth positioning data, removing localization ambiguities.

In order to generate the EXPECTEDOG occupancy grids associated

[2]These annotations are available at: http://www.ira.disco.unimib.it/intersection-ground-truth

(a) Frame: 0000000122; Distance: 7.28011 m; Crossing Topology: 7; In Crossing: true

(b) Frame: 0000000000; Distance: 10.4403 m; Crossing Topology: 7; In Crossing: false

(c) Frame: 0000000393; Distance: 21.0238 m; Crossing Topology: 6; In Crossing: false

(d) Frame: 0000000086; Distance: 32.7567 m; Crossing Topology: 1; In Crossing: false

Figure 5.11: Some examples from the proposed intersection groundtruth, ordered by distance from the center. The figure labels show: frame number, distance from crossing center (from OpenStreetMap, considering the RTK), crossing topology, in crossing flag (false/true). We set the intersection topologies as soon as the vehicle is 30 meter or less from the crossing center (as detected from OpenStreetMap).

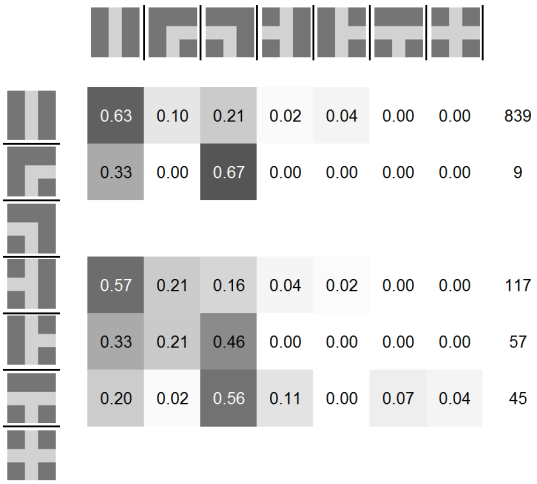
Figure 5.12: The EXPECTEDOG are generated considering all the hypotheses shown in Figure 5.10. Considering the topology and the w and r parameters of the nearest intersection, the remainder of the topologies are then modified by applying the retrieved information.

with the different intersection hypotheses, we leveraged the OpenStreetMap service by retrieving the geometry of the closest intersection along the traveling direction, *i.e.*, the c, n, w_i and r_i parameters of the intersection model proposed in Chapter 4 and shown in Figure 4.2a. Using these parameters, we sample 6 new hypotheses as to complete the 7 pattern candidates shown in Figure 5.10 by means of the following scheme. The expected occupancy grids are generated by spatially sampling the intersection hypotheses, *i.e.*, considering the w and r geometric properties of the nearest intersection extracted from the OpenStreetMap service and applying it to all the 7 topologies that we take in consideration. This expedient allows us to maintain a limited number of hypotheses, yet considering all the 7 patterns in the evaluation process. An example of the sampled EXPECTEDOGs is shown in Figure 5.12.
After their generation, a score is associated to each EXPECTEDOG by means of the template matching procedure proposed in Section 4.6, allowing the detector to identify the intersection topology. The results of the experimental activities are summarized in the form of the confusion matrices shown in Figure 5.14, which compares how many times each topology has been correctly classified with respect to all the frames where such topology appears in the sequence (indicated at the end of each row). In all the experiments we evaluated the nearby intersections only if it was totally visible, therefore we never took into account the frames where the *in crossing* flag was set to true.

In Figures 5.14d to 5.14f we show the overall confusion matrices of all the meaningful sequences, except for the difficult sequence shown in Figure 5.13. To better stress the importance of the achievable results,

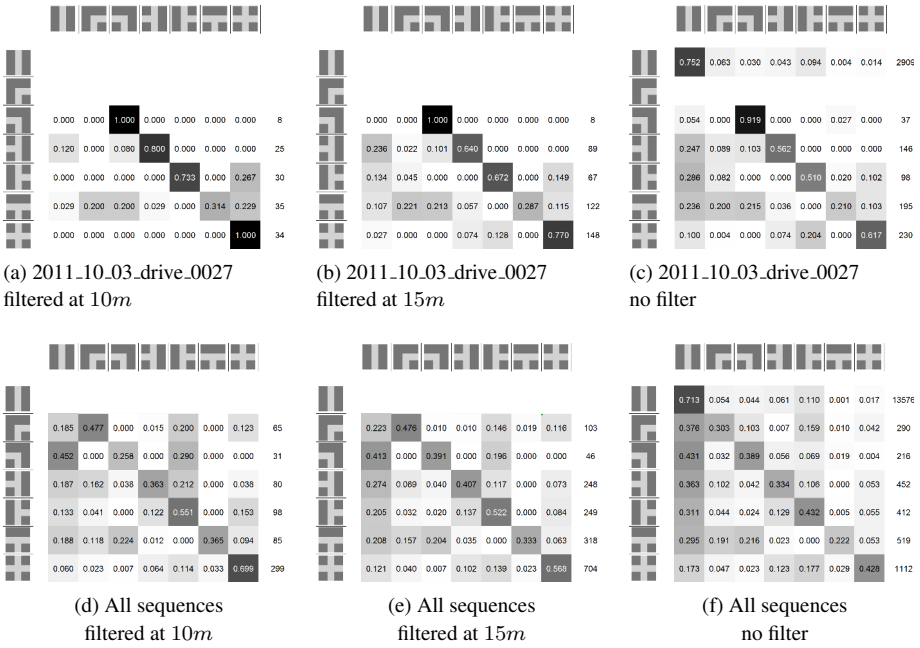
(a)

(b)



(c)

Figure 5.13: In Figure 5.13c the resulting confusion matrix after the evaluation on the difficult 2011_09_30_drive_0034 sequence. A frame from the sequence containing a brick-paved road is shown in Figure 5.13a, while Figure 5.13b contains an uneven surface. In such a situation, evaluating the intersection hypotheses unavoidably leads to a wrong estimate. Please note that the *right* topology that was always misinterpreted as *left* contains only 9 uphill roads (last column outside the matrix).



(a) 2011_10_03_drive_0027 filtered at $10m$

(b) 2011_10_03_drive_0027 filtered at $15m$

(c) 2011_10_03_drive_0027 no filter

(d) All sequences filtered at $10m$

(e) All sequences filtered at $15m$

(f) All sequences no filter

Figure 5.14: The confusion matrices show the results after the evaluation on the used KITTI sequences. Each row ends with the total frames number of each intersection configuration. In the first row we show the results using the $2011\_10\_03\_drive\_0027$ sequence, filtering the intersection topology detections using $10m$ or $15m$ as to evaluate the detections while gradually increasing the detection difficulty (the third matrix includes the complete sequence). In the same way, the second row summarizes the results of the approach over all the used KITTI sequences shown in Table 5.3. For a comparison with the state-of-the-art results, we refer the reader to Figure 5.15.

in Figures 5.14a to 5.14c we present the results on the longest sequence 2011_10_03_drive_0027.

In Figures 5.14a and 5.14d, we filtered the results as to hold only the frames taken at less than 10 meters from the intersection, whereas in Figures 5.14b and 5.14e the threshold value was set as to include images taken up to 15 meters and in Figures 5.14c and 5.14f we took into account all the sequence (without any limitation).

5.3.2 Image Classification Assessment

Our semantic segmentation pipeline hinges on two classification steps. As described in Section 4.3, following the TextonBoost approach proposed by Shotton *et al.* [62], the first classification step relies on a per-pixel analysis (which creates the so-called unary potentials) supported by a second step refinement phase, by means of a CRF approach. With

(a) SUTSU (b) GIST

Figure 5.15: For a comparison with the state-of-the-art results, we refer inserted here the confusion matrices of the SUTSU and GIST algorithms presented in [21]. Please note that we do not consider the *place* class, *i.e.*, we do not recognize whether the vehicle is inside the intersection.

respect to the CRF approach, we used the PyStruct inference library proposed by Mueller [203], using in all the training and validation phases the dataset sequences proposed in Section 4.3.1 together with the k-fold cross validation technique. In order to evaluate the per-pixel unary potentials, which are then used within the CRF approach, we train the multi-feature variant of the TextonBoost algorithm proposed in the work of Ladický *et al.* [204], which creates a *texton map* vector by evaluating Location, Color, Histogram of Oriented Gradients (HOG) and the 17-dimensional filter-bank suggested in [62] (an example of the features is shown in Figure 5.16). The quantitative evaluation of the achieved performance was obtained using the standard Precision, Recall and F_1 measures defined as follows:

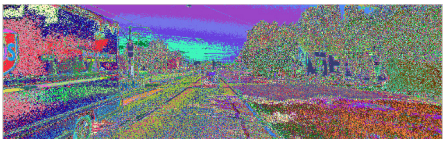
$$precision = \frac{TP}{TP+FP} \tag{5.1}$$

$$recall = \frac{TP}{TP+FN} \tag{5.2}$$

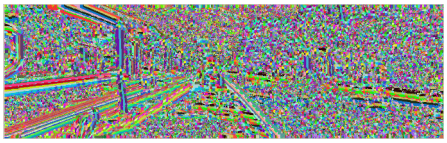
$$F_1 = \frac{2TP}{2TP+FP+FN} \tag{5.3}$$

We achieved satisfying results for the unary classification phase, with a good F_1 value equals to 0.93 for the road surface class (other results are summarized in the first row of Table 5.6). After the per-pixel evaluation, we trained the CRF model varying both the inference algorithm as well as the pairwise configuration schemes. According to the PyStruct documentation, we changed the CRF configuration modifying the following parameters:

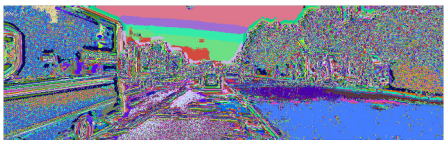
- the Superpixel algorithm,



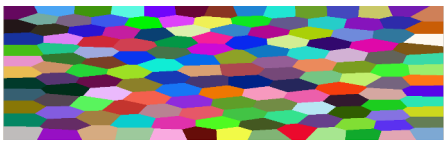
(b) Color Textons



(c) HOG Textons



(d) Filter-bank Textons



(e) Location Textons

Figure 5.16: Texton maps evaluated on a frame from the KITTI dataset.

- the Edges connection model (*i.e.*, the feature vector used to weight the spatial dependency between the superpixels),
- the Symmetric and Antisymmetric Edge properties,
- the composition of the Vector used to weight the superpixel dependencies, accordingly to Table 5.5.

An exhaustive evaluation of the capabilities of the proposed CRF configuration was performed by means of the 17 tests proposed in Table 5.4 and the results are presented in Table 5.6. The parameters highlighted has proven to be slightly more effective with respect to the others.

The Figure 5.17 shows the qualitative results of the unary classification along with each CRF refinement, with respect to the *road*, *sidewalk* and *other* classes. As shown in the figure, the spatial consistency constraint of the CRF provides smooth sidewalk classifications, decreasing the typical scattering effect arising from per-pixel algorithm. However, as can be noticed in the third and sixth columns of the Table 5.6, the spatial coherence introduced with the CRF did not give the expected performance gain with respect to the road classification task. In particular, we have noticed that the image disparity information did not introduce enhancements in the CRF classification. Given the good results reported in

Table 5.4: In the following 17 tests, we evaluate the CRF classification results modifying the underlying structure configuration.

Test	Superpixel Algorithm	Vector Code	Symmetric Edge	Antisymm Edge	Connect. Scheme	Inference Algorithm
1	SLIC	1	no	no	Pairwise	qpbo
2	SLIC	2	no	no	Pairwise	qpbo
3	SLIC	3	no	no	Pairwise	qpbo
4	SLIC	4	no	no	Pairwise	qpbo
5	SLIC	3	yes	no	Pairwise	qpbo
6	SLIC	3	yes	yes	Pairwise	qpbo
7	SLIC	3	no	yes	Pairwise	qpbo
8	SLIC	3	yes	yes	3-Extend	qpbo
9	SLIC	3	yes	yes	Pairwise	ad3
10	quickshift	3	yes	yes	Pairwise	qpbo
11	SLIC-zero	3	yes	yes	Pairwise	qpbo
12	quickshift	no	no	no	Pairwise	qpbo
13	quickshift	5	yes	yes	Pairwise	qpbo
14	quickshift	5	yes	yes	3-Extend	qpbo
15	quickshift	5	yes	yes	10-Extend	qpbo
16	quickshift	5	yes	yes	5-Extend	qpbo
17	quickshift	3	yes	yes	3-Extend	qpbo

Regarding the superpixel segmentation, we tested the following methods: the Quickshift [205], SLIC [206] and SLIC-zero variant, which are are available within the Python scikit-image [207] library. The details of the feature code value, representing the superpixel dependency vector (*i.e.*the values associated to the edges between superpixels, in the pairwise scheme), is shown in Table 5.5. Finally, the highlighted row indicates the best configuration. Surprisingly, this configuration does not leverage the disparity information in the edge scheme.

Table 5.5: Vector Codes

Vector Code	Contrast Function	Superpixel Distance	Disparity	Orientation
1	1	no	no	yes
2	2	no	no	yes
3	1	yes	no	yes
4	2	yes	no	yes
5	1	yes	yes	yes

In the table, the composition of the five testing feature vectors associated to the edges. Regarding the Contrast function, we evaluated both the algorithms available within the PyStruct library. Disparity and Contrast are referred to the average of all pixels in the superpixel.

Table 5.6: In this table we show the evaluation measures of both the Unary potentials (first row) and the CRF configurations.

Test	Road			All Class		
	Precision	Recall	F1	Precision	Recall	F1
Unary	0.91	0.96	0.93	0.96	0.96	0.96
Test n.1	0.89	0.96	0.93	0.96	0.95	0.95
Test n.2	0.89	0.96	0.93	0.96	0.95	0.95
Test n.3	0.90	0.96	0.93	0.96	0.95	0.95
Test n.4	0.90	0.96	0.93	0.96	0.95	0.95
Test n.5	0.90	0.96	0.93	0.96	0.95	0.95
Test n.6	0.89	0.96	0.93	0.96	0.95	0.95
Test n.7	0.89	0.96	0.93	0.96	0.95	0.95
Test n.8	0.89	0.96	0.93	0.96	0.95	0.95
Test n.9	0.89	0.96	0.93	0.96	0.95	0.95
Test n.10	0.90	0.95	0.93	0.96	0.95	0.95
Test n.11	0.90	0.95	0.93	0.96	0.95	0.95
Test n.12	0.90	0.96	0.92	0.96	0.95	0.95
Test n.13	0.90	0.95	0.93	0.96	0.95	0.95
Test n.14	0.89	0.96	0.92	0.96	0.95	0.95
Test n.15	0.90	0.95	0.92	0.96	0.95	0.95
Test n.16	0.89	0.96	0.92	0.96	0.95	0.95
Test n.17	0.90	0.95	0.93	0.96	0.96	0.96

In this table, the results of both the per-pixel (highlighted row) and CRF (Test 1-17) classifications by means of the standard Precision Recall and F_1 measures.

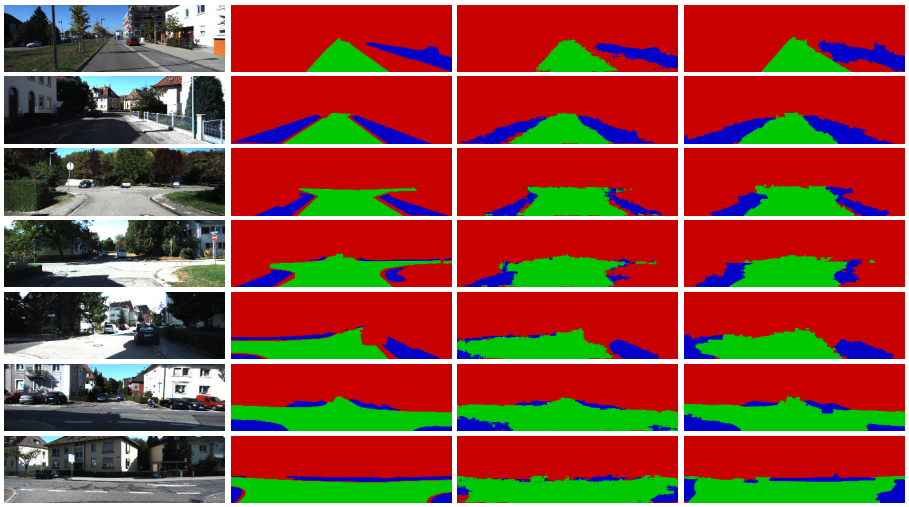


Figure 5.17: The Figure compares the classification accuracy of the unary potentials with respect to the CRF approach. From left to right: image from the on-board camera, ground-truth classification, unary classification, CRF classification.

Table 5.7: The experimental activity configuration.

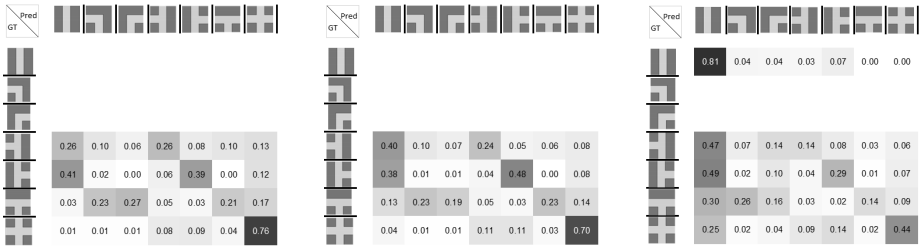
Test Number	Occupancy Grid Scheme
1	PCL Only
2	CRF Only
3	PCL and CRF method integrated using Dempster-Shafer
4	PCL and CRF method integrated using PCR6

the recent works by Sengupta *et al.* [68], we believe that the superpixel configuration employed within the CRF model has negatively affected the effectiveness of the method, overwhelming the per-pixel classification. In spite of these limitations, the next section describes how we can still exploit each of these classifications to tackle the road intersection detector scheme.

5.3.3 Geometric vs Semantic Intersection Classification

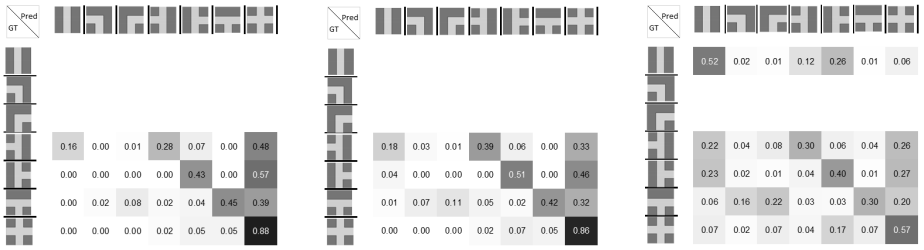
Differently from the per-pixel classifications shown in the previous section, here the task is related to the understanding of the intersection topology. In order to detect the geometric configuration of the roads, our approach exploits a bird-eye-view representation of the surrounding road surface, detected using the algorithms described in Sections 4.2 and 4.3. Following the scheme proposed in Table 5.7, here we present the individual results of the aforementioned algorithms, as well as their integration using the Dempster-Shafer Theory and the PCR6 Rule. We assessed the performances exploiting the residential 2011_10_03_drive_0027 sequence, which contains the highest number of intersections in its category.

On the one hand, according to the results shown in Figures 5.18 and 5.19, the geometric approach tends to better classify the intersection topologies with respect to the semantic classification but it also favors a detection even on straight roads, mainly because of the poor curbs identification and lack of 3D reconstruction precision. On the other hand, the CRF approach frequently classifies intersections areas as straight road, and presents a performance asymmetry with respect to the Type-2 and Type-3 roads (left or right junctions). From a technical perspective, we believe that these issues stem from the training dataset, which does not contain a representative and balanced set of intersections. In spite of our best efforts, we have not found any good training dataset specifically designed for the intersection detection evaluation, as the best datasets like [68, 198] include a small subset of intersection images. Even though



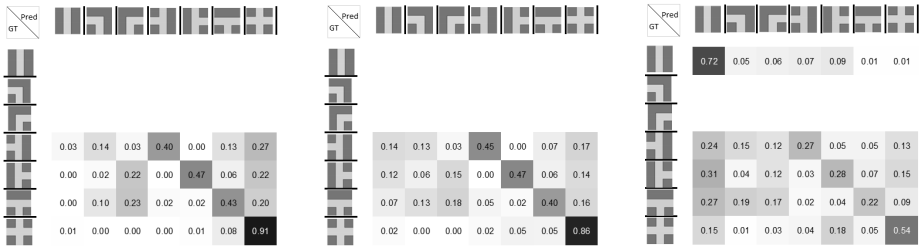
(a) Filtered at 10 m (b) Filtered at 15 m (c) All dataset

Figure 5.18: **Test 1**. CRF only, with no temporal integration



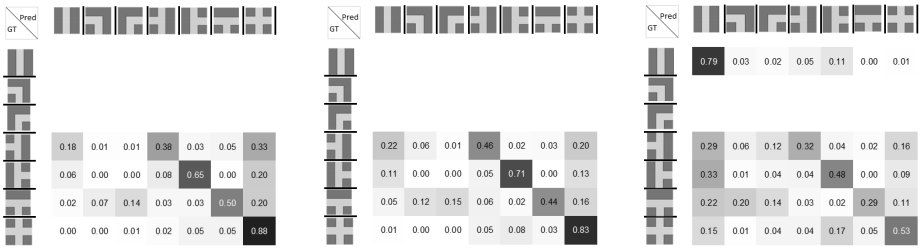
(a) Filtered at 10 m (b) Filtered at 15 m (c) All dataset

Figure 5.19: **Test 2**. PCL only, with no temporal integration



(a) Filtered at 10 m (b) Filtered at 15 m (c) All dataset

Figure 5.20: **Test 3**. PCL and CRF using Dempster-Shafer, with no temporal integration



(a) Filtered at 10 m (b) Filtered at 15 m (c) All dataset

Figure 5.21: **Test 4**. PCL and CRF using PCR6, with temporal no integration

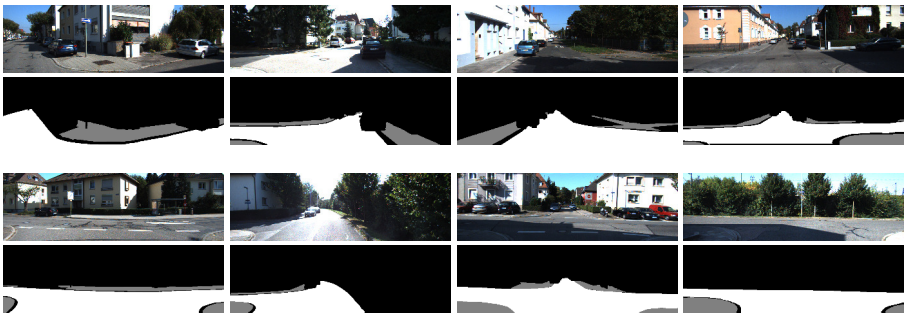


Figure 5.22: We manually annotated extra images in order to obtain a larger and more representative training set of intersection areas. In the figure, an example of the intersection ground-truth. The dataset can be downloaded from http://www.ira.disco.unimib.it/iralab/intersection-detector/

we introduced a new set of manually annotated images in the training phase (containing intersection areas, see Figure 5.22), the spatial coherence results did not give a performance gain in road classification. Despite these critical considerations, the combination of geometric with semantic classification outperformed each individual classification in almost all the experiments. Regarding the integration schemes between the two classifications, we have tested both the Dempster-Shafer and the PCR6 rule proposed in the previous chapter. The results, shown in Figures 5.20 and 5.21, shows that the PCR6 rule allows us to better integrate conflicting information with respect to the Dempster-Shafer rule.

The overall scheme, presented in Sections 4.4 and 4.5 leverages the aforementioned results.

5.3.4 Temporal integration

Both the geometric and semantic approaches shown before do not take into consideration any time coherence scheme. In order to reduce the negative effect of unstable detections, and differently from the CRF temporally coupling strategy proposed by Floros *et al.*in [101], here we propose a temporal integration by means of consecutive occupancy grids integration.

As shown in Figure 4.10, the overall scheme consists of three process:

- A first integration component related to the integration of the individual PCLOG and CRFOG, (shown in Figure 5.23).
- A temporal hysteresis grid of the classified values, so that multiple

same-class classifications increase the classification belief in both the CRFOG and PCLOG.

- A final fusion of both the PCLOG and the CRFOG along with the corresponding hysteresis grids.

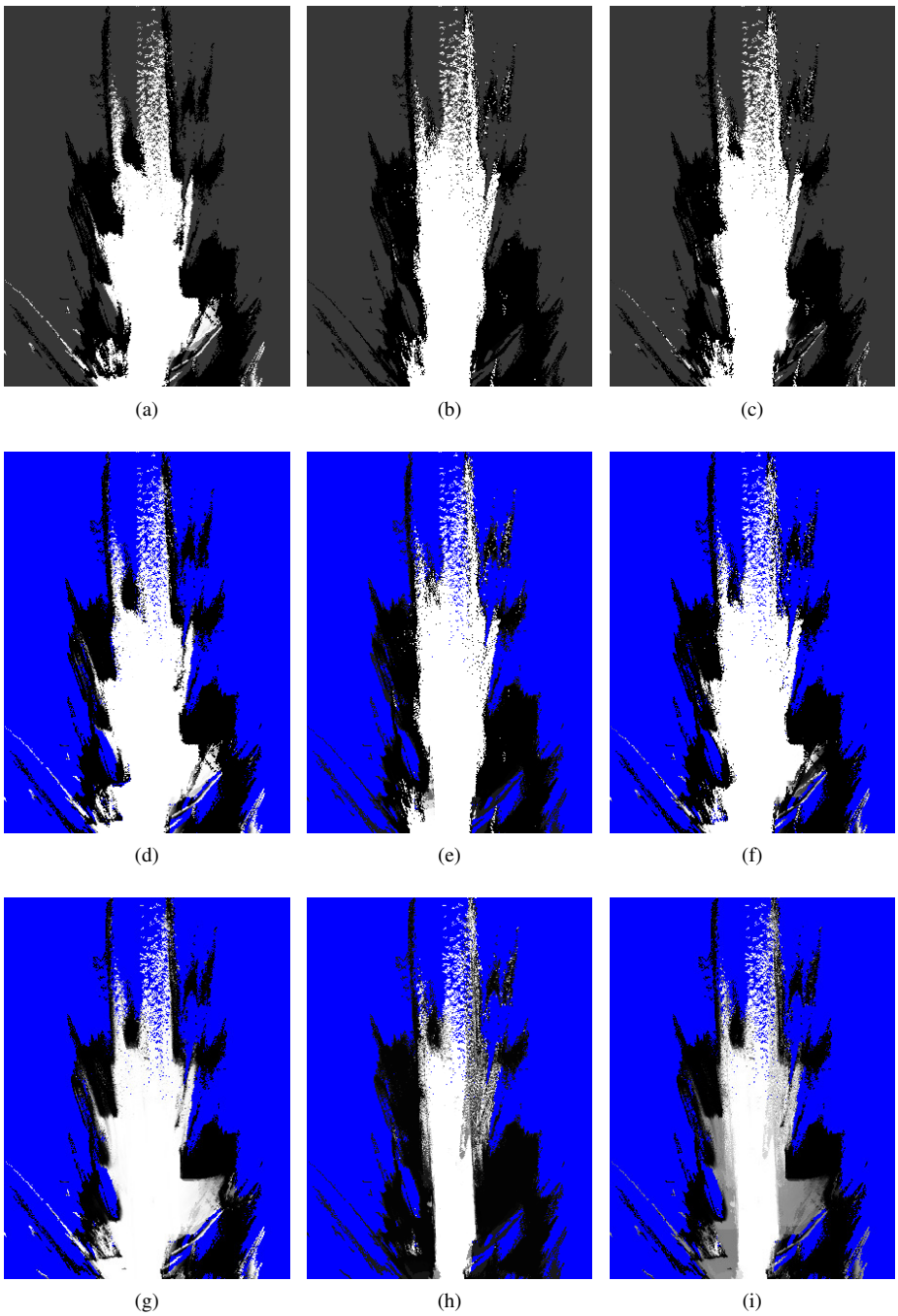
The results of our temporal integration procedure are shown in Figure 5.24. In comparison with the results previously shown in Figure 5.14, the temporal procedure introduce a recognition improvement, that become more meaningful as the vehicle approaches the intersection, as can be seen in the confusion matrices at 15 and 10 meters.

5.3.5 Scoring Function Assessment

The classification algorithm described in Section 4.6 evaluates the best intersection topology through a template matching procedure, considering the detected SENSORSOGs and the hypothesized EXPECTEDOGs. To validate the overall procedure, we initially assess the scoring function capabilities with respect to the distance of the intersection center, exploiting the sequence 2011_10_03_drive_0027 For this purpose, starting from a couple of grids as depicted in Figure 5.26, we slightly move the mask (*i.e.*, the EXPECTEDOG) longitudinally towards the crossing center, as a way to verify where the maximum value of the scoring function is achieved when the two grids are aligned. This is by no means trivial, since the reconstruction noise and cluttering elements, together with the road visibility issues, make the recognition process hard. Using this procedure, we can now assess the maximum distance from which the crossing can be classified. In Figure 5.25 we show the results of considering four different Type-7 intersections at 11, 14, 19 and 24 meters from the vehicle. The x-axis reports the distances at which the EXPECTEDOG was moved while the y-axis the associated score. According to the results, the scoring function shows satisfactory performances up to 15 m from the intersection and barely sufficient at 18 m.

5.3.6 Discussion

Although the proposed on-line approach showed very good results for most of the tested sequences, and, to the best of our knowledge, achieved comparable or better performance with respect to the other comparable approaches in the literature, we observed some treacherous sequences

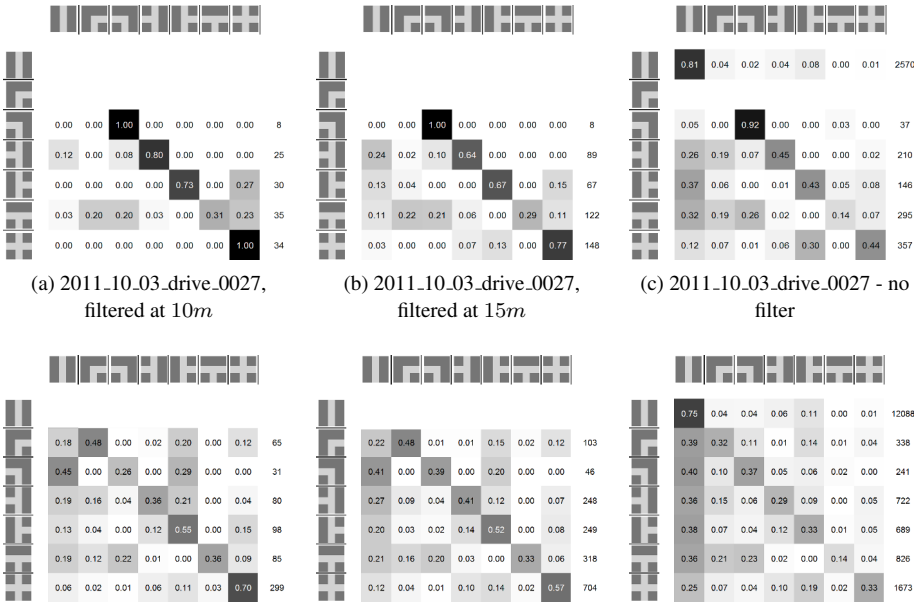


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 5.23: In this figure we show the results of the different temporal integration schemes, ordered by achieved performances. The second raw depicts the Dempster-Shafer method and the third row the same integration using the PCR6 rule. For the sake of completeness, in the first row we report also the temporal integration using the Bayes scheme, which was definitely not used. Please notice the strong negative impact on the reconstruction performances, particularly visible in the last column.



(a) 2011_10_03_drive_0027, filtered at $10m$

(b) 2011_10_03_drive_0027, filtered at $15m$

(c) 2011_10_03_drive_0027 - no filter

(d) All sequences, filtered at $10m$

(e) All sequences, filtered at $15m$

(f) All sequences - no filter

Figure 5.24: As in Figure 5.14, the resulting confusion matrices after the evaluation on different KITTI sequences, with the temporal integration enabled.

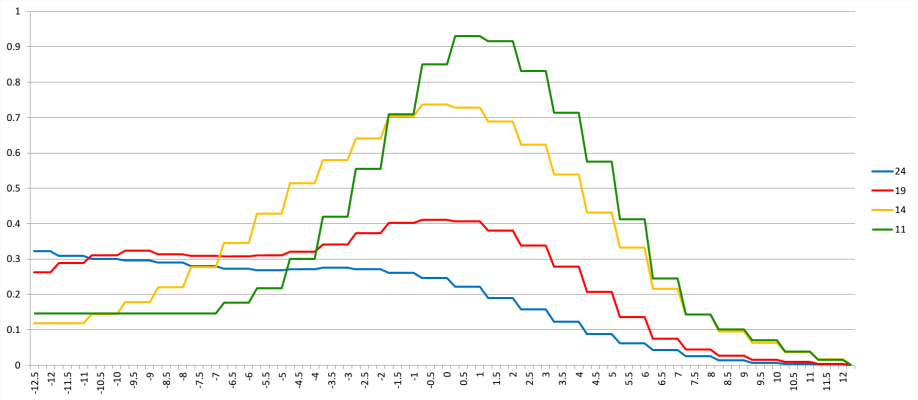
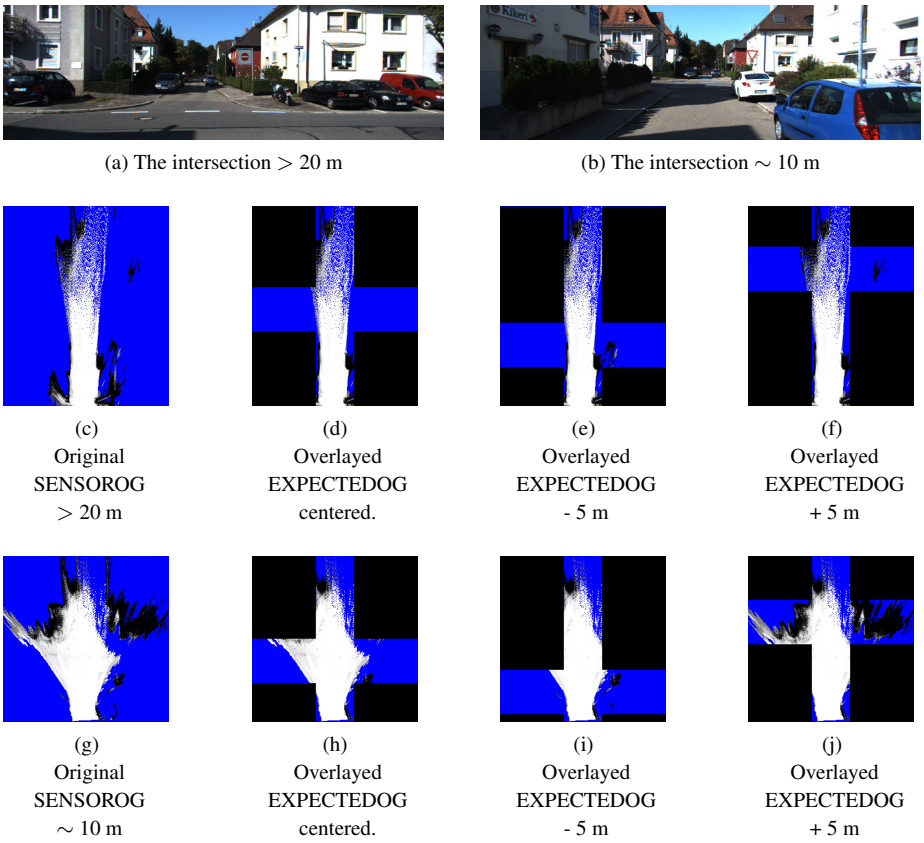


Figure 5.25: We evaluated the scoring function at different distances, sliding the EXPECTEDOG from the true position to verify the resulting score. As it can be seen considering the blue line, the function does not always allow us to distinguish the intersection, and its reliability drops about 15 to 20 meters.



(a) The intersection $>$ 20 m

(b) The intersection $\sim$ 10 m

(c) Original SENSOROG $>$ 20 m

(d) Overlayed EXPECTEDOG centered.

(e) Overlayed EXPECTEDOG - 5 m

(f) Overlayed EXPECTEDOG + 5 m

(g) Original SENSOROG $\sim$ 10 m

(h) Overlayed EXPECTEDOG centered.

(i) Overlayed EXPECTEDOG - 5 m

(j) Overlayed EXPECTEDOG + 5 m

Figure 5.26: In Figures 5.26a and 5.26b, the same intersection at the approximately distance from the vehicle of 20 m and 10 m. In Figures 5.26c and 5.26g the related SENSOROGs used for the scoring function assessment, along with three different overlayed EXPECTEDOGs. In each of the three sequences, the mask was set to the center of the crossing, then shifted by +5 and -5 meters longitudinally towards the crossing center.

that led to misleading results. These sequences include scenes with low image and geometric contrast, with respect to the road boundaries, including both geometric and visual clues. For this reason, we consider that further researches should include other specific road features in the CRF model, aiming at leveraging the road spatial relations that in this work we tried to include with the superpixel approach.

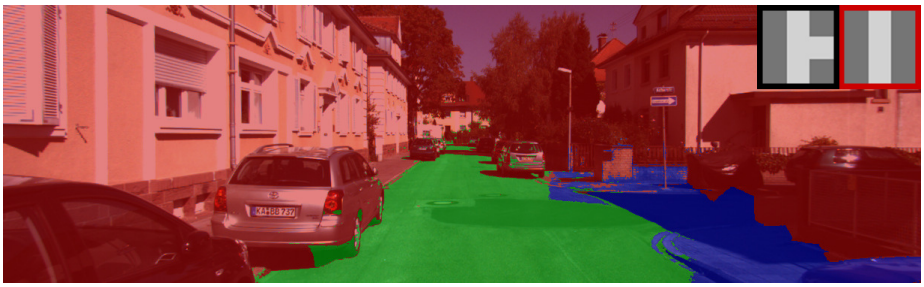
Despite this limitations, as it can be seen from the confusion matrices in Figures 5.18 to 5.21, the best detections were achieved synergically exploiting the results coming from the integration of the geometric and semantic pipelines. Moreover, when considering the temporal consistency between consecutive frames introduced in Section 5.3.4, the ap-

proach shows a good robustness with respect to the individual yet noisy occupancy grids and the topology classification task.

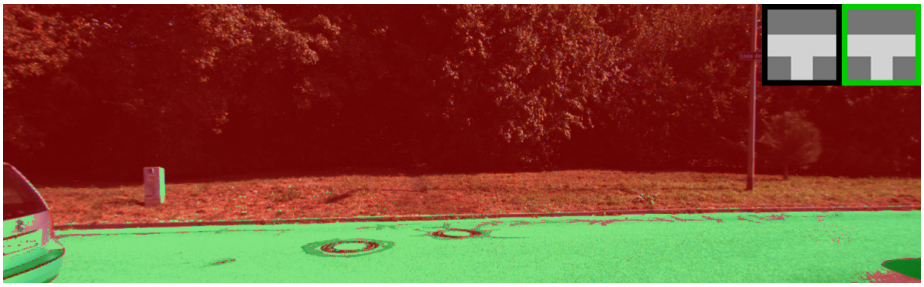
In Figure 5.13c we provided an example where the proposed approach was not able to cope with the classification goal, which is part of the difficult 2011_09_30_drive_0034 sequence. Considering this last assessment, it is important to note that this sequence represents an extremely challenging scenario, as it includes brick-paved roads whilst no images of brick-paved were available for the training phase.

In Figures 5.27 and 5.28 we show some results evenly divided by three successfully and three erroneous detections. As it is shown, the detector achieved the best performance when the road boundaries are well defined. On the other hand, in uphill and downhill scenarios, as well as in strong shadows, occlusions dominate in the perceived image. Despite these considerations, we consider that including further crossing configuration as well sloping roadways in the training database should allow the imaging pipeline to better cope with these circumstances. Finally, in Figure 5.29, we show an array of consecutive frames taken from the final evaluation phase on the 2011_10_03_drive_0027 sequence. Here we stress the fact the temporal consistency scheme allows us to deal with the unstable CRF detections, as can be noticed with the more steady occupancy grids and the consequent detected topology.

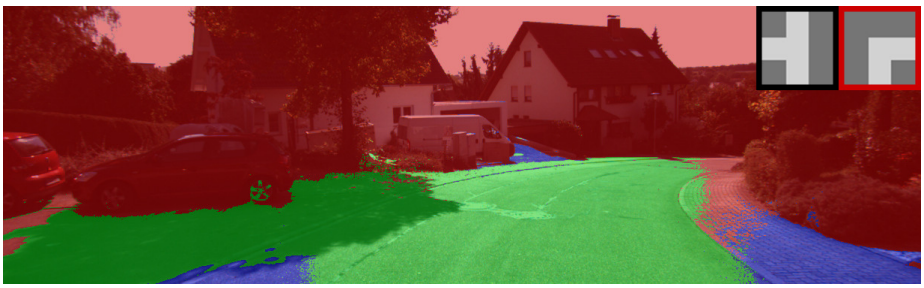
(a)



(b)



(c)



(d)

Figure 5.27: Some examples of the topology reconstruction of our system. On the upper right of every figure the is reported the real topology (black border) alongside with the inferred topology (green border for correct prediction, and red for incorrect). Figures 5.27a, 5.27c and 5.28a: correctly inferred topology. Figures 5.27b, 5.27d and 5.28b: respectively wrong topology inferred due to strong shadows, downhill roads and occlusions. See Figure 4.1 for color coding.

(a)

(b)

Figure 5.28: Some examples of the topology reconstruction of our system. On the upper right of every figure the is reported the real topology (black border) alongside with the inferred topology (green border for correct prediction, and red for incorrect). Figures 5.27a, 5.27c and 5.28a: correctly inferred topology. Figures 5.27b, 5.27d and 5.28b: respectively wrong topology inferred due to strong shadows, downhill roads and occlusions. See Figure 4.1 for color coding.

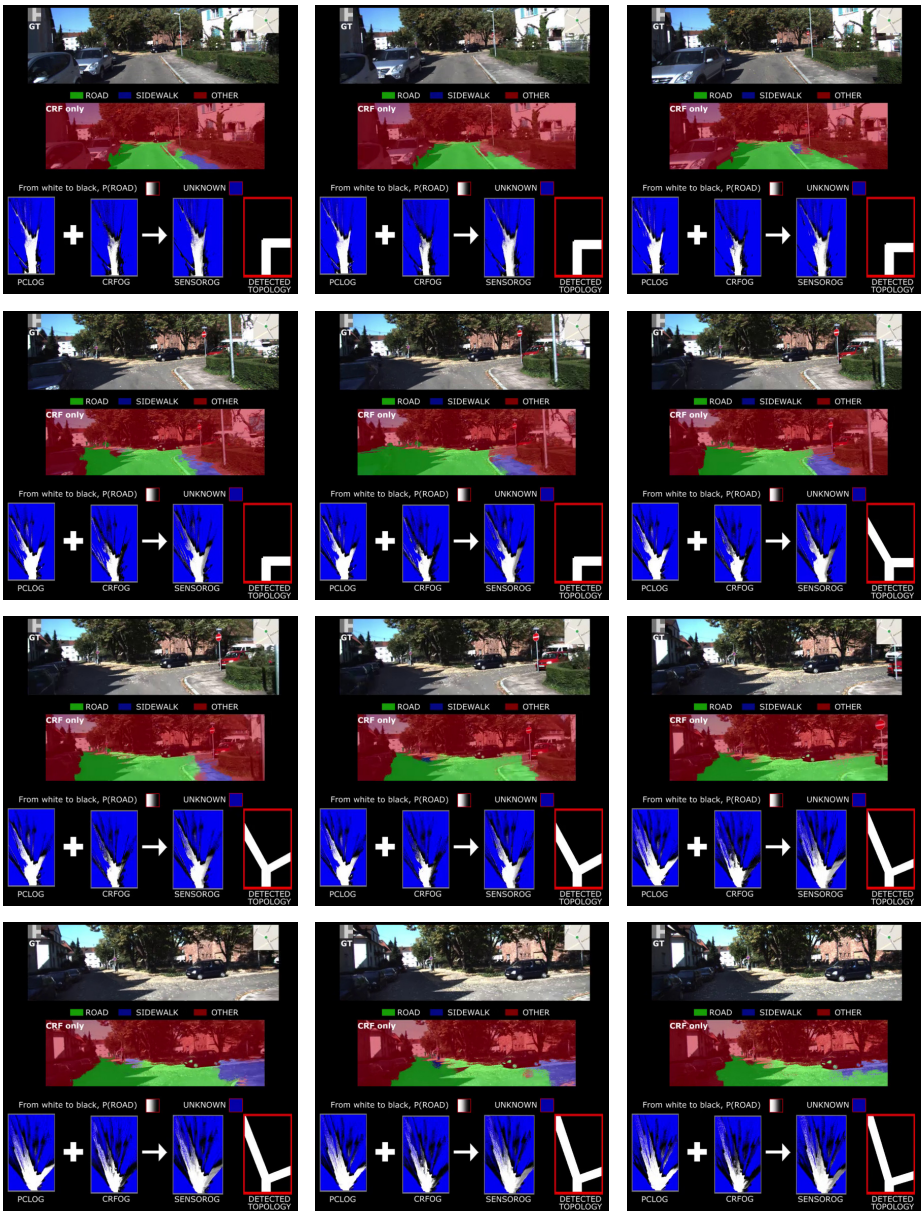


Figure 5.29: In the images, the intersection classification results as the vehicle approaches an intersection (consecutive frames). In each sub-image, the first image is the vehicle's camera left, with the ground truth and the position images overlayed in the upper corners. The second image represents the CRF classification only. As can be noticed, the sidewalk is frequently classified as road in successive frames. The temporal consistency scheme allows us to deal with these unstable detections, as can be noticed with the more steady occupancy grids. The results of the basic PCLOG and CRFOG as well as their integration in the SENSOROG is shown in the three images in the last line, together with the predicted intersection topology.

5.4 Lane and Lines - Highway case study

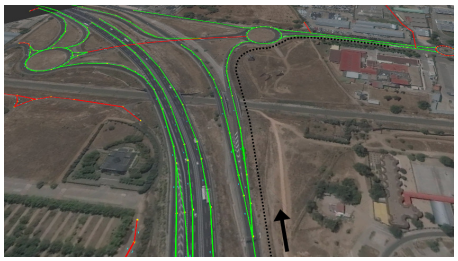
In this session we show the preliminary results achieved by leveraging the two Layout Components presented in Section 3.4. The proposed algorithms tackle the localization task within highway environments, leveraging and extending the results achieved by means of the aforementioned OpenStreetMap Matching Pipeline. The approach was verified exploiting an ad-hoc dataset collected in real driving conditions using the Drivertive autonomous vehicle (shown in Figure 5.30a) of the INVETT Research group of the University of Alcalá. The dataset was taken driving on the A-2 highway area in Espartales Norte, Alcalá de Henares, Spain.

5.4.1 Road Width Component

The experimental activity was performed by leveraging the Road Layout Framework in the area depicted in Figure 5.30b. Besides the different dataset, we used a configuration similar to the one proposed in Section 5.1. The only difference was related to the hypotheses initialization phase. Here the latter were initialized considering a narrow uncertainty area, in order to avoid any localization failure resulting from a wrong initial lock-on-road procedure. As depicted in Figure 5.31a, the most common issues regarding the lock-on-road localization procedures arises in proximity of Y-junctions. The reason is twofold: on the one hand, due to the lack of in-lane accuracies, the system needs to artificially injected error as to ensure a proper representation of the feasible state space. On the other hand, straight roads does not allow the framework to leverage the structure of the road graph to reduce the longitudinal uncertainties resulting from the unavoidable accumulated drift of the Visual Odometry (VO) systems (in this case, the LIBViso2 library). This effect, in combination with even a slight error of the VO, may result in a wrong lock-on-road result. We assessed the localization enhancements over our previous approach by counting the number of hypotheses locked on each OpenStreetMap segment as the vehicle traveled on the A-2 highway, measuring the number of required frames to have the whole hypotheses set centered on the correct road segment.

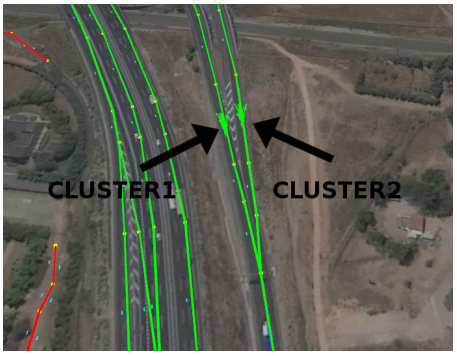
we have a semantic gap in the lane number as the vehicle travels the part of the road with an extra lane

(a) The Drivertive Vehicle used to collect the Highway dataset

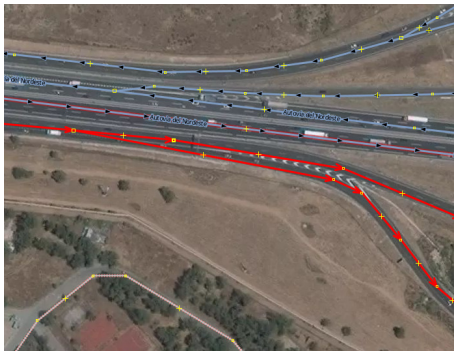


(b) Testing Site - A2 Highway. During the Road With Layout Component evaluation, the vehicle turns onto the exit ramp that run along the main highway.

Figure 5.30



(a)



(b)

Figure 5.31: The figure depicts a typical Y-junction. Since the ramp follows the route of main road, there is no chance for the OpenStreetMap component to distinguish between the two clusters.

Discussion

As can be noticed in Figures 5.32 and 5.33, the first approach, relying on the OpenStreetMap component only, tends to hold the localization on the wrong road segment. This effect can be explained by the initial misalignment of the hypotheses with respect to the direction of the junction. For instance, let us consider the case depicted in Figure 5.34, just before the vehicle turns onto the exit ramp. As can be seen in the image on the left, the highway and the ramp share the highlighted node, resulting in a different direction of the first part of the ramp. According to the equations shown in Section 3.2.2, it follows that the hypotheses aligned with the main highway trunk will be better considered by our framework. On the one hand, increasing the misalignment threshold of the OpenStreetMap Layout component could represent a quick trivial solution, which delivers good results at the cost of worst localization performances with

respect to the cartographic map. According to the OpenStreetMap documentation[3], an alternative solution for the problem would be to use the *Open Ways*, *i.e.*, *"a way describing a linear feature which does not share a first and last node"*, as depicted in Figure 5.34b. However the latter is not a viable option, due to the de facto *Shared Node* solution commonly used by OpenStreetMap community.

On the other hand, quantitative the results show that the component proposed in this section allow the framework to quickly respond to the lane change, once the straight lines are detected. In Figures 5.32e and 5.32f we report two suboptimal result of the component. In the first case, it should be notice that despite the hypotheses cover both the road segments, the framework failed to lock on the right highway trunk. On the other hand, the second graph shows the recovery from failures ability introduced with the new layout component.

5.4.2 Road Lane Component

The road width components has proved to enhance the localization accuracy as the vehicle is driven in close proximity to parallel roads, but it do not introduce any enhancement towards the in-lane localization. In this section we show the preliminary results achieved by leveraging the module shortly presented in Section 3.4.3. Here we exploited the same Alcalá dataset, considering the sequence shown in Figure 5.35, which is composed of 513 frames and 4 lane transitions.

The goal of the proposed model is to estimate the vehicle's ego-lane, given the position of the detected road markings. The model is designed to be robust enough to tolerate the noisy measurements resulting from the basic line detector presented in Section 3.4.1. For this purpose, we leveraged a Hidden Markov Model approach. From a technical perspective, the HMM model implements a filtering procedure over a single discrete random variable. Here, the state space X_t is defined on the number of possible states, in this case the number of available lanes, *i.e.*, 3, representing the probability of being in one of the three lanes. These multiple state variables are combined in a single "megavariable" whose tuples are all possible tuples of values of the individual state variables, as describe in [186]. To perform the necessary predictions, a transition model was empirically derived by evaluating the performances of the model. The resulting transition matrix is shown in Table 5.8. The prediction is

[3] http://wiki.openstreetmap.org/wiki/Way

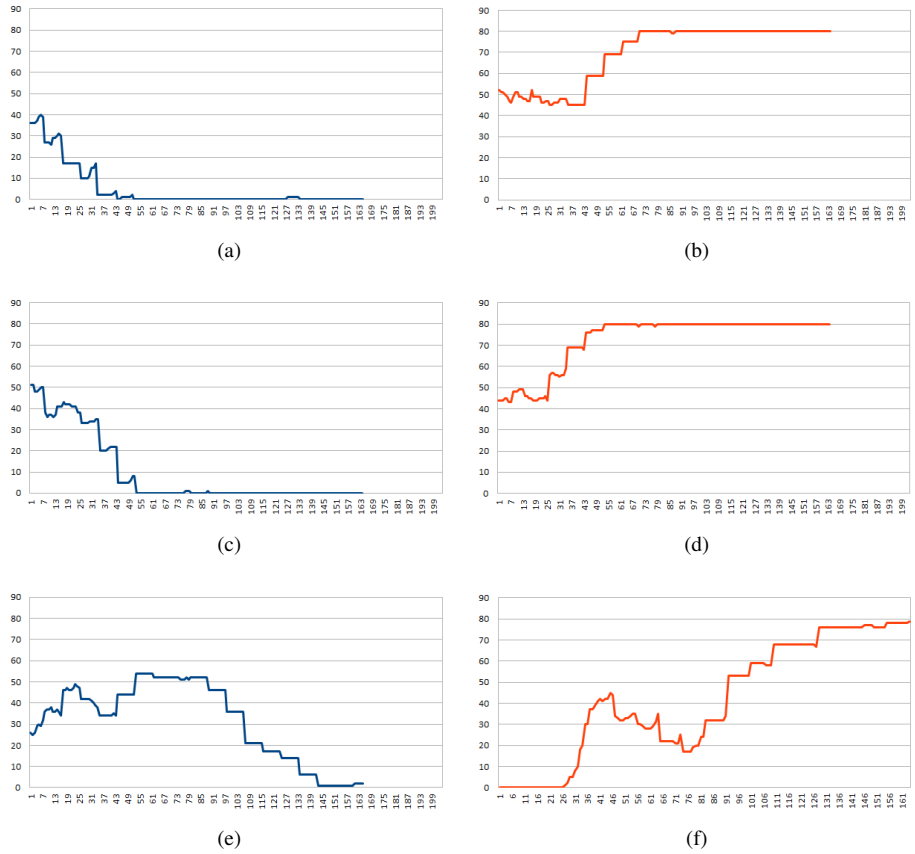
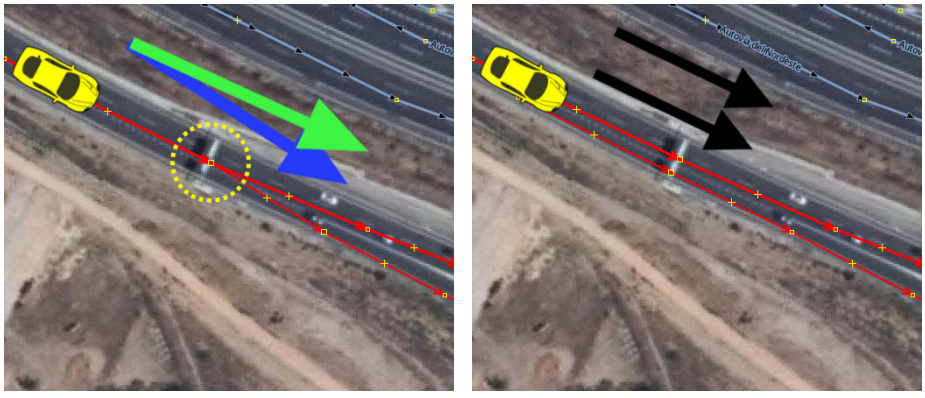


Figure 5.32: In this figure, we compare some of the results achieved using the proposed Road Width Component (red graphs on the right) with respect to the original proposal presented in Section 5.1 (blue graphs on the left). In both cases the vertical axis represents the number of hypotheses locked on the correct road segment. The total number of hypotheses was set to 80. All the sequences end after 20 seconds from the starting point, resulting in 200 width evaluations. The starting point was set as the vehicle approaches the exit ramp, *i.e.*, just before the Y-junction shown in Figure 5.34.

Table 5.8: Transition Matrix

0.63	0.279	0.001	0.0869	0.003	0.0001
0.139	0.63	0.139	0.003	0.086	0.003
0.001	0.279	0.63	0.0001	0.003	0.0869
0.1519	0.027	0.0001	0.567	0.253	0.001
0.023	0.054	0.023	0.208	0.484	0.208
0.0001	0.027	0.1519	0.001	0.253	0.567

Figure 5.33: In the figure we show the starting point and the end point of the evaluation phase shown in Figure 5.32. The average threshold represents the average point where the proposed Layout Component correctly identified the correct road trunk.



(a) (b)

Figure 5.34: In the figure, a common Y-junction. This configuration may lead the system to snap into the wrong highway trunk.

(a) (b) (c) (d)

Figure 5.35: In the figure, the four lane transitions performed while capturing the dataset.

evaluated according to the previous state multiplied by the transition matrix. With regards to the measurement and the associated measurement model, let us consider again the Figure 3.18. First, for each lane, we create likelihood counter, for instance L_1, L_2, L_3. Then, we estimate the vehicle ego-lane by iterating the following considerations over all the detected lines, *i.e.*, valid and not valid:

- if the lane distance is compatible with the ego-lane, and the line has the continuous flag, we add 1 to the L_1 and L_3 counters.
- if the lane distance is not compatible with the ego lane, we add 1 to the each counter which is in accordance with the measurement. As an example, considering the highlighted line in Figure 5.36, we add 1 to L_2 and L_3 counters, as the distance is in accordance with both the lanes.

Regarding the probability values of the sensor model, we evaluate its reliability by leveraging the line *counter* as introduced in Section 3.4.1. The resulting L_1, L_2, L_3 counters, *i.e.*, our measure, along with the given sensor reliability and the transition matrix, allows us to filter the X_t space state representing the current ego-lane belief of our model. For a thorough review of the Hidden Markov Models, we refer the reader to [186].

Discussion

This last component was designed to tackle the noisy measurements resulting from INVETT line detector. The quantitative results of the algorithm performances are summarized in Figure 5.37. On the one hand, and not surprisingly, the results show that the line detector is unable to correctly detect the number of lanes. As depicted in Figures 5.37b, 5.37e and 5.37h, the detector results are extremely noisy, resulting in an unreliable ego-line detection. As instance, the detector is completely missing

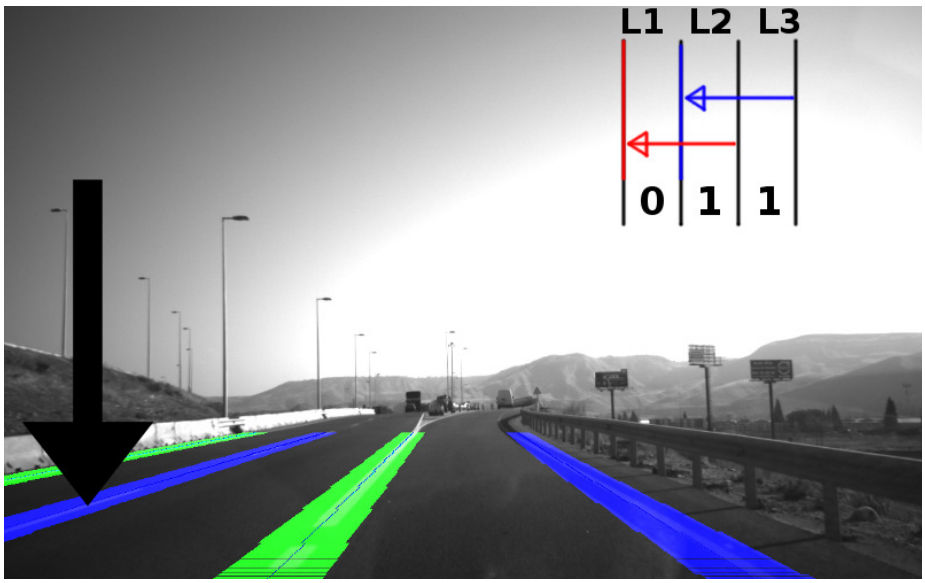


Figure 5.36: Considering the line indicated with the arrow, we can estimate that the probability of being Lane$_{\{1|2|3\}}$ is $\{0,0.33,0.33\}$. This vector, represents the HMM "megavariabile" X_t. This procedure is evaluated for all the lines in the detection.

Table 5.9: HMM vs Naive Detector Ego-Lane Estimate

	Lane 1	Lane 2	Lane 3	Sum of Errors	Fault Rate
Detector Failures	91	216	25	364	0.70
HMM Failures	37	75	76	188	0.36
Ground Truth frames in lines	219	216	82		

the transitions from Lane2 to Lane1, from Lane1 to Lane2 and from Lane2 to Lane3. On the other hand, the filtering effect of the HMM model is clearly shown in Figures 5.37a to 5.37i. Here the proposed model correctly identified the lane transitions, and promising results are summarized in Table 5.9. With respect to the experimental activity and the results, it is clear that even with a slightly better line detector would result in a great improvement.

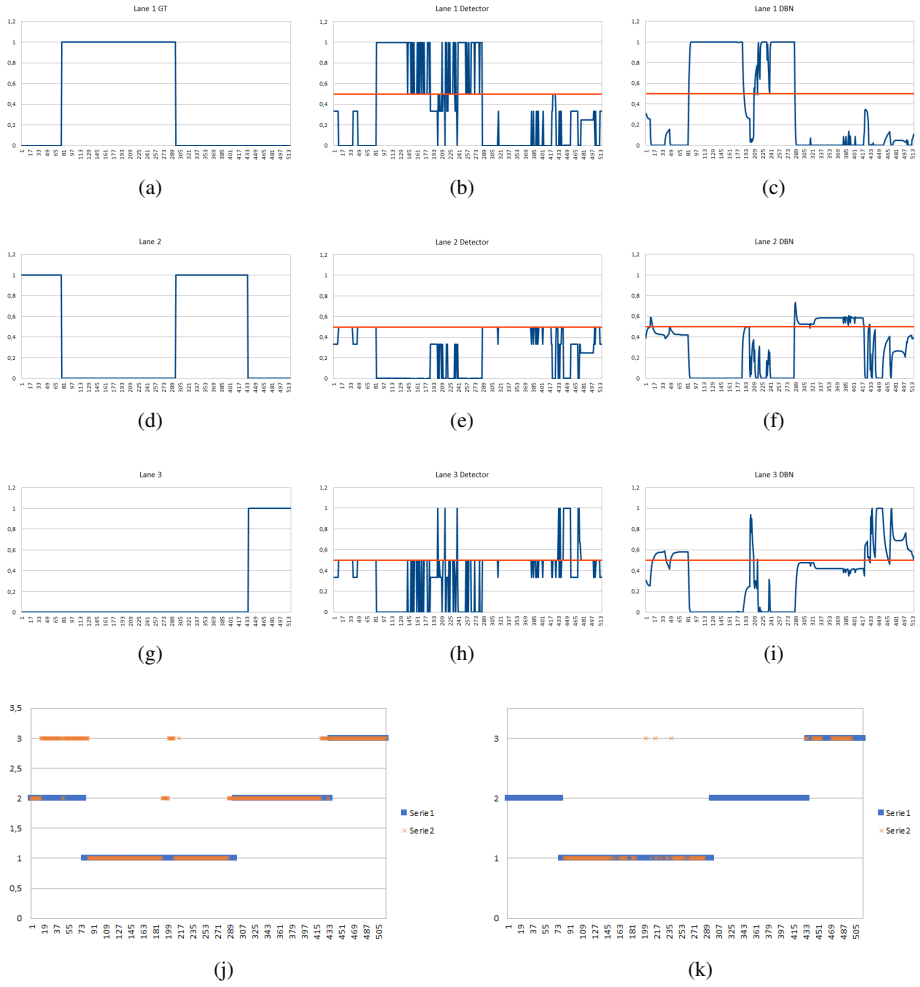


Figure 5.37: In the figure, the results from the evaluation of the Road Lane Component. The first part of the figure shows the probabilities associated to each of the three lanes shown in Figure 5.36. The first column represents the ground truth with respect to the vehicle's ego-lane (manually annotated). The second column outputs the ego lane estimation as results from the detector, *i.e.*, without the proposed HMM model, in a per-frame basis. In the third column we show the results of the the proposed approach. In second part of the figure, we show the most likely ego-lane value evaluated with and without the proposed model. In both graphs the ground truth (Serie 1) is highlighted in blue, while the estimated lane (Serie 2) is shown in orange. As can be noticed, the results of the proposed model leads to more stable and accurate ego-lane estimates.

5.5 Conclusion

In this chapter, we have presented the results of the four sensing pipelines introduced within this thesis. Three out of four pipelines were actively used within the Road Estimation framework proposed in Chapter 3, introducing valuable performances in the context of both urban and highway vehicle localization. The main outcomes are:

- The first evaluation of the framework was performed with the map matching module, which allows us to tackle in real-time a multi hypotheses localization problem by leveraging the open source LIBViso2 visual odometry software and the OpenStreetMap service. Relying on topometric maps as the only global information to perform vehicle driving localization may lead to indiscernible situations. On the other hand, our interpretation led to excellent results, outperforming the state-of-the-art works in both accuracy performances and quantitative assessment.
- After the analysis of the weak points of the map matching module, to better handle the treacherous scenarios resulting from misalignment and coarse road approximations frequently present in current topometric maps, we present the results of our Building Detection pipeline, aimed at enhancing the initial rough road-level localization achieved with the first component. The results shown enhancements in terms of both lateral and longitudinal localization by smoothing the trajectories obtained by the original algorithm, *i.e.*, by leveraging the OpenStreetMap road network graph only.
- With respect to our on-line intersection detector, the evaluation activity was focused on the testing of the classification engine. The latter is composed of a twofold scheme that includes, on the one hand, a per-pixel classification followed by a refinement phase by means of a Conditional Random Field approach and on the other hand a PCL-based purely geometric classification. Even though the results from this pipeline were not integrated in the Road Layout Estimation framework, we had the opportunity to stress the performances with respect to state-of-the-art algorithms in the road intersection field, achieving very good results despite the not always reliable detection pipeline. This considerations allowed us to have this work accepted as a contribution to the International Conference on Robotics and Automation - ICRA2017. More importantly,

our method has a separate sensing pipeline, with respect to the intersection topology classification. Unlike other more sophisticated systems, which exploit a sequence of images up to when the vehicle is inside the intersection to perform an off-line scene understanding task, we developed an on-line template matching scheme that relies on a versatile intersection model able to leverage the prior information obtained from the OpenStreetMap service.

- Finally, we present the experimental activity resulting from two specific road features aimed ad enhancing the localization in highway scenarios. The proposed approaches have been tested under real traffic conditions, showing satisfactory performances with respect to the map-matching-only settings and compensating the noisy measures of our basic line detector.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

This thesis has presented a probabilistic framework aimed at estimating the ego-vehicle localization in both urban and highway scenarios. We tackled the problem proposing an on-line framework designed to handle the localization uncertainties by means of component-wise interaction with the OpenStreetMap mapping service. In this section we summarize the main contributions of this thesis:

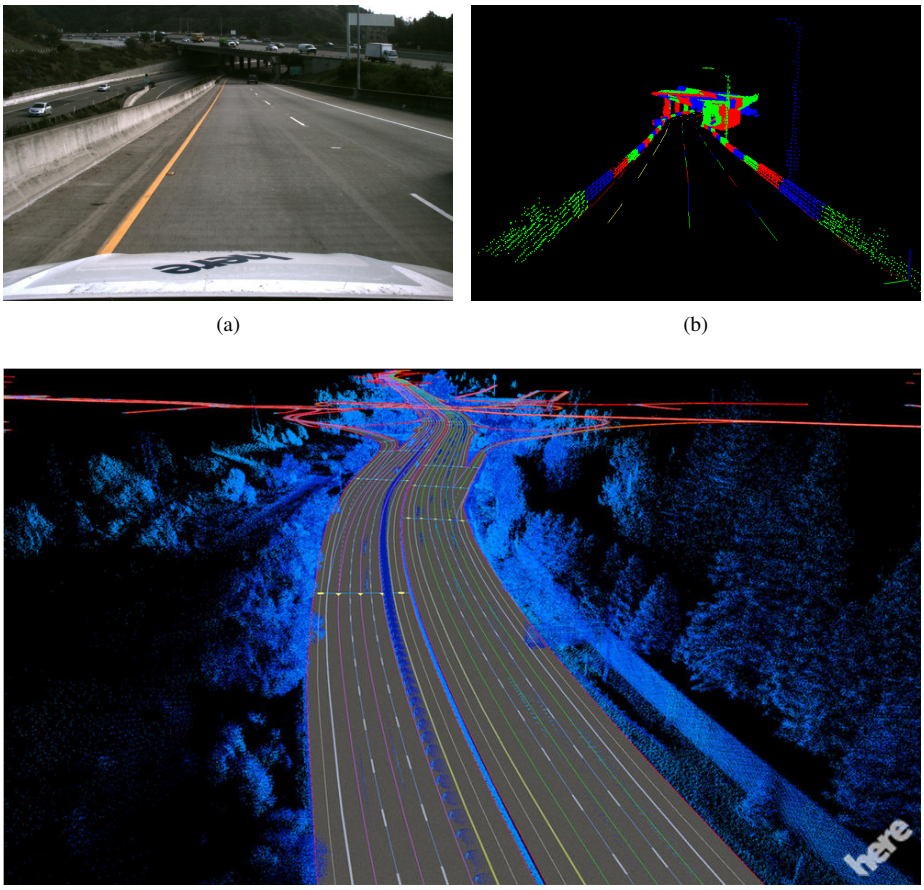
- The localization task is critical for every autonomous system and, in the context of intelligent transportation systems, even a slightly erroneous position estimate could have a strong impact on the system safety. However, with respect to the standard robot localization problem, the road vehicle localization has specific characteristics that should be taken into account. Our claim is that standard topological maps could introduce remarkable added value for vehicle localization. To deal with the critical nature of the autonomous vehicles, we have proposed a probabilistic approach to match the heterogeneous outcomes from sensing pipelines against the OpenStreetMap. The main insights of the proposed approach is its on-line strategy and its flexibility with respect to the number of input sources. This work has been presented in the 18th International Conference on Intelligent Transportation Systems 2015.
- We have presented a method aimed at lowering the accuracy limitations that can arise in GNSS denied urban-like environments, by means of a building's façades detection pipeline. In addition

to the road matching with data from the OpenStreetMap service, building's outlines help the framework to achieve in-lane localization performance in urban areas. This work has been presented in the 19th International Conference on Intelligent Transportation Systems 2016.

- We have proposed an innovative module to on-line tackle the intersection modeling problem, which take into account 3D geometry and visual clues as well as temporal integration between measurements. This was our first step towards the semantic analysis of the scene. To the best of our knowledge, our system has achieved state-of-the-art performances with respect to similar on-line approaches, classifying the road topology by means of dual classification, geometry and pixel-level.
- We have shown how the OpenStreetMap road properties like lane numbers and road width, coupled with Hidden Markov Models and a very basic line tracker, can be exploited to achieve a richer awareness of the ego-vehicle position.

6.2 Future Works

This theses showed how the existing cartographic maps can be exploited also by perception algorithms. There are several future directions that should be considered. On the one hand, the Layout Components presented in this work have paved the way for further feature integrations. Road signs, traffic lights, parking lots are just some examples of useful features that can be leveraged for localization purposes. Also dynamic objects such as other vehicles or pedestrians would help our system in the localization process. Another interesting direction will be to update, validate or even integrate the features within the OpenStreetMap service. Furthermore, it should be considered that the latest works leveraging Convolutional Neural Networks and Deep Learning recently outperformed the state-of-the-art segmentation algorithms like the Conditional Random Fields. Integrating this techniques together with the proposed detection pipelines would result in a direct improvement. As a final consideration, we want to stress how the next generation of upcoming high-definition maps are going to change the current map-matching capabilities. In Figure 6.1 we show an example of these maps.



(a) (b)

(c)

Figure 6.1: An real example of the next generation high definition maps containing both 3D laser scanner data and semantic annotations. Even if this are quite simple scenarios in which the proposed framework

Bibliography

[1] SAE International. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles, 2016.

[2] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J. Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, Michael Sokolsky, Ganymed Stanek, David Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: Systems and algorithms. *2011 IEEE Intell. Veh. Symp.*, pages 163–168, jun 2011.

[3] J.C. McCall and M.M. Trivedi. Video-Based Lane Estimation and Tracking for Driver Assistance: Survey, System, and Evaluation. *IEEE Trans. Intell. Transp. Syst.*, 7(1):20–37, mar 2006.

[4] Ernst D. Dickmanns and Birger D. Mysliwetz. Recursive 3-D Road and Relative Ego-State Recognition, 1992.

[5] Mohamed Aly. Real time detection of lane markers in urban streets. In *2008 IEEE Intell. Veh. Symp.*, pages 7–12. IEEE, jun 2008.

[6] Radu Danescu and Sergiu Nedevschi. Probabilistic Lane Tracking in Difficult Road Scenarios Using Stereovision. *IEEE Trans. Intell. Transp. Syst.*, 10(2):272–282, jun 2009.

[7] Gaoya Cao, Florian Damerow, Benedict Flade, Markus Helmling, and Julian Eggert. Camera to Map Alignment for Accurate Low-Cost Lane-Level Scene Interpretation. *Proc. IEEE Intell. Transp. Syst. Conf.*, (November):in press, 2016.

[8] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. Recent progress in road and lane detection: a survey. *Mach. Vis. Appl.*, 25(3):727–745, apr 2014.

[9] Ignacio Parra Alonso, David Fernández Llorca, Miguel Gavilan, Sergio Álvarez Pardo, Miguel Ángel Garcia-Garrido, Ljubo Vlacic, and Miguel Ángel Sotelo. Accurate Global Localization Using Visual Odometry and Digital Maps on Urban Environments. *IEEE Trans. Intell. Transp. Syst.*, 13(4):1535–1545, dec 2012.

[10] J Laneurit, C Fouque, and G Dherbomez. MULTI-HYPOTHESIS MAP-MATCHING USING. In *16th World Congr. ITS Syst. Serv.*, pages 1–8, 2009.

[11] Georgios Floros, Benito van der Zander, and Bastian Leibe. OpenStreetSLAM: Global vehicle localization using OpenStreetMaps. *2013 IEEE Int. Conf. Robot. Autom.*, pages 1054–1059, may 2013.

[12] Hernan Badino, Daniel Huber, and Danfei Xu. Topometric Localization on a Road Network. In *IROS*, 2014.

[13] M Hentschel and B Wagner. Autonomous robot navigation based on OpenStreetMap geodata. *Intell. Transp. Syst. (...*, pages 1645–1650, 2010.

[14] Jeffrey a. Delmerico, Philip David, and Jason J. Corso. Building facade detection, segmentation, and parameter estimation for mobile robot localization and guidance. In *2011 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1632–1639. IEEE, sep 2011.

[15] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Van Gool, and W. Purgathofer. A survey of urban reconstruction. *Comput. Graph. Forum*, 32(6):146–177, 2013.

[16] Dorra Larnaout, Vincent Gay-Belllile, Steve Bourgeois, and Michel Dhome. Vision-Based Differential GPS: Improving VSLAM / GPS Fusion in Urban Environment with 3D Building Models. *2014 2nd Int. Conf. 3D Vis.*, pages 432–439, 2014.

[17] Mayank Bansal and Kostas Daniilidis. Geometric Urban Geolocalization. In *2014 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3978–3985. IEEE, jun 2014.

[18] Yanlei Gu, Li-Ta Hsu, and Shunsuke Kamijo. GNSS/On-Board Inertial Sensor Integration with the Aid of 3D Building Map for

Lane-Level Vehicle Self-Localization in Urban Canyon. *IEEE Trans. Veh. Technol.*, 9545(c):1–1, 2015.

[19] Philipp Ruchti, Bastian Steder, Michael Ruhnke, and Wolfram Burgard. Localization on OpenStreetMap data using a 3D laser scanner. In *2015 IEEE Int. Conf. Robot. Autom.*, pages 5260–5265. IEEE, may 2015.

[20] Dissertation Dipl and Andreas Geiger. *Probabilistic Models for 3D Urban Scene Understanding from Movable Platforms*. PhD thesis, 2013.

[21] Andreas Ess, Tobias Mueller, Helmut Grabner, and Luc Van Gool. Segmentation-Based Urban Traffic Scene Understanding. *Proceedings Br. Mach. Vis. Conf. 2009*, pages 84.1–84.11, 2009.

[22] Charles E. Thorpe. SCARF: A Color Vision System that Tracks Roads and Intersections. *IEEE Trans. Robot. Autom.*, 9(1):49–58, 1993.

[23] V. Gengenbach, H.-H. Nagel, F. Heimes, G. Struck, and H. Kollnig. Model-based recognition of intersections and lane structures. In *Proc. Intell. Veh. '95. Symp.*, pages 512–517. IEEE, 1995.

[24] Christopher Rasmussen. Road shape classification for detecting and negotiating intersections. *IEEE Intell. Veh. Symp. Proc.*, pages 422–427, 2003.

[25] Jie Du, Jason Masters, and Matthew Barth. Lane-level positioning for in-vehicle navigation and automated vehicle location (AVL) systems. In *Proceedings. 7th Int. IEEE Conf. Intell. Transp. Syst. (IEEE Cat. No.04TH8749)*, pages 35–40. IEEE, 2004.

[26] E.D. Dickmans. Subject-object discrimination in 4D dynamic scene interpretation for machine vision. In *[1989] Proceedings. Work. Vis. Motion*, pages 298–304. IEEE Comput. Soc. Press, 1989.

[27] T. Jochem, D. Pomerleau, B. Kumar, and J. Armstrong. PANS: A Portable Navigation Platform. *Intell. Veh. '95 Symp.*, pages 107–112, 1995.

[28] Takeo Kanade and Charles E Thorpe. CMU strategic computing vision project report : 1984 to 1985. 1986.

[29] Charles Thorpe, Martial H. Hebert, Takeo Kanade, and Steven A. Shafer. Vision and Navigation for the Carnegie-Mellon Navlab., 1988.

[30] Massimo Bertozzi and Alberto Broggi. GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Process.*, 7(1):62–81, 1998.

[31] Alberto Broggi, Massimo Bertozzi, and Alessandra Fascioli. ARGO and the MilleMiglia in Automatico Tour. *IEEE Intell. Syst.*, 14(1):55–64, jan 1999.

[32] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The 2005 DARPA grand challenge: the great robot race*, volume 36. Springer Science & Business Media, 2007.

[33] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the DARPA Grand Challenge. *J. F. Robot.*, 23(9):661–692, sep 2006.

[34] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[35] Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, Doug Johnston, Stefan Klumpp, Dirk Langer, Anthony Levandowski, Jesse Levinson, Julien Marcil, David Orenstein, Johannes Paefgen, Isaac Penny, Anna Petrovskaya, Mike Pflueger, Ganymed Stanek, David Stavens, Antone Vogt, and Sebastian Thrun. Junior: The stanford entry in the urban challenge. *Springer Tracts Adv. Robot.*, 56(October 2005):91–123, 2009.

[36] Erico Guizzo. How google's self-driving car works, 2011.

[37] Sebastian Thrun. The google blog - what we're driving at, 2010.

[38] European Commission. Towards a European road safety area: policy orientations on road safety 2011-2020. *Framework*, pages 1–15, 2010.

[39] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[40] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *Proc. Natl. Conf. Artif. Intell.*, (Handschin 1970):343–349, 1999.

[41] Dieter Fox. KLD-sampling: Adaptive particle filters. *Adv. Neural Inf. Process. Syst. 14*, 14(1):713–720, 2002.

[42] C. Kwok, D. Fox, and M. Meila. Adaptive real-time particle filters for robot localization. *2003 IEEE Int. Conf. Robot. Autom. (Cat. No.03CH37422)*, 2:2836–2841, 2003.

[43] Rudolph Triebel, Patrick Pfaff, and Wolfram Burgard. Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing. In *2006 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2276–2282. IEEE, oct 2006.

[44] M. Herbert, C. Caillas, E. Krotkov, I.S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. *Proceedings, 1989 Int. Conf. Robot. Autom.*, pages 997–1002, 1989.

[45] Simon Lacroix, Anthony Mallet, David Bonnafous, G. Bauzil, Sara Fleury, Matthieu Herrb, and Raja Chatila. Autonomous Rover Navigation on Unknown Terrains: Functions and Integration. *Int. J. Rob. Res.*, 21(10-11):917–942, oct 2002.

[46] Rainer Kümmerle, Rudolph Triebel, Patrick Pfaff, and Wolfram Burgard. Monte Carlo localization in outdoor terrains using multilevel surface maps. *J. F. Robot.*, 25(6-7):346–359, jun 2008.

[47] R. Kummerle, D. Hahnel, Dmitri Dolgov, Sebastian Thrun, and Wolfram Burgard. Autonomous driving in a multi-level parking structure. In *2009 IEEE Int. Conf. Robot. Autom.*, pages 3395–3400. IEEE, may 2009.

[48] A L Ballardini, A Galbiati, M Matteucci, F Sacchi, and D G Sorrenti. An effective 6DoF motion model for 3D-6DoF Monte Carlo Localization. *Work. Planning, Percept. Navig. Intell. Veh.*, pages 4–9, 2012.

[49] Ian Baldwin and Paul Newman. Road vehicle localization with 2D push-broom LIDAR and 3D priors. *Proc. - IEEE Int. Conf. Robot. Autom.*, pages 2611–2617, 2012.

[50] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: an efficient probabilistic 3D mapping framework based on octrees. *Auton. Robots*, 34(3):189–206, apr 2013.

[51] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. Map-Based Precision Vehicle Localization in Urban Environments. *Robot. Sci. Syst. III*, pages 121–128, 2008.

[52] here.com. Here hd live map, 2016.

[53] here.com. How map based adas can make you a better driver, 2016.

[54] Augusto Luis Ballardini, Simone Fontana, Axel Furlan, Dario Limongi, and Domenico Giorgio Sorrenti. A Framework for Outdoor Urban Environment Estimation. In *2015 IEEE 18th Int. Conf. Intell. Transp. Syst.*, pages 2721–2727. IEEE, sep 2015.

[55] Marvin Raaijmakers and Mohamed Essayed Bouzouraa. In-vehicle Roundabout Perception Supported by A Priori Map Data. *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC*, 2015-Octob:437–443, 2015.

[56] Theo Gevers and Felipe Lumbreras. Combining Priors, Appearance, and Context for Road Detection. 15(3):1168–1178, 2014.

[57] Jose M Alvarez, Theo Gevers, and Antonio M Lopez. 3D Scene Priors for Road Detection. pages 57–64, 2010.

[58] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Recovering Surface Layout from an Image. *Int. J. Comput. Vis.*, 75(1):151–172, jul 2007.

[59] Jiangye Yuan and Anil M. Cheriyadat. Road Segmentation in Aerial Images by Exploiting Road Vector Data. In *2013 Fourth Int. Conf. Comput. Geospatial Res. Appl.*, pages 16–23. IEEE, jul 2013.

[60] Gellert Mattyus, Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Enhancing Road Maps by Parsing Aerial Images Around the World. In *2015 IEEE Int. Conf. Comput. Vis.*, volume 11-18-Dece, pages 1689–1697. IEEE, dec 2015.

[61] Philipp Bender, Julius Ziegler, and Christoph Stiller. Lanelets: Efficient Map Representation for Autonomous Driving. (Iv), 2014.

[62] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-class Object Recognition and Segmentation. In *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, volume 3951 LNCS, pages 1–15. 2006.

[63] Christian Wojek and Bernt Schiele. A Dynamic Conditional Random Field Model for Joint Labeling of Object and Scene Classes. pages 733–747, 2008.

[64] Sanjiv Kumar and Hebert. Discriminative random fields: a discriminative framework for contextual interaction in classification. In *Proc. Ninth IEEE Int. Conf. Comput. Vis.*, pages 1150–1157 vol.2. IEEE, 2003.

[65] S Kumar and M Hebert. Man-made structure detection in natural images using a causal multiscale random field. In *2003 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, 2003. Proceedings.*, pages I–119–I–126. IEEE Comput. Soc, 2003.

[66] Stephen Gould, Richard Fulton, and Daphne Koller. Decomposing a scene into geometric and semantically consistent regions. *2009 IEEE 12th Int. Conf. Comput. Vis.*, (Iccv):1–8, 2009.

[67] Ruiqi Guo and Derek Hoiem. Beyond the Line of Sight: Labeling the Underlying Surfaces. In *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, volume 7576 LNCS, pages 761–774. 2012.

[68] Sunando Sengupta, Eric Greveson, Ali Shahrokni, and Philip H. S. Torr. Urban 3D semantic modelling using stereo vision. *2013 IEEE Int. Conf. Robot. Autom.*, pages 580–585, may 2013.

[69] Fernando Lafferty, John and McCallum, Andrew and Pereira. An Introduction to Conditional Random Fields. *Proc. eighteenth Int. Conf. Mach. Learn. ICML*, 1:282–289, 2001.

[70] Axel Furlan. *Robotic Perception for Autonomous Navigation*. PhD thesis, Università degli Studi di Milano - Bicocca.

[71] Nico Cornelis, Bastian Leibe, Kurt Cornelis, and Luc Van Gool. 3D urban scene modeling integrating recognition and reconstruction. *Int. J. Comput. Vis.*, 78(2-3):121–141, 2008.

[72] Ashutosh Saxena, Min Sun Min Sun, and Andrew Y a.Y. Ng. Learning 3-D Scene Structure from a Single Still Image. *ICCV 3dRR-07*, 31(5):824–840, 2007.

[73] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *2009 IEEE 12th Int. Conf. Comput. Vis.*, number Iccv, pages 1849–1856. IEEE, sep 2009.

[74] Paul Sturgess, Karteek Alahari, Lubor Ladicky, and Philip H. S. Torr. Combining Appearance and Structure from Motion Features for Road Scene Understanding. *Procedings Br. Mach. Vis. Conf. 2009*, pages 62.1–62.11, 2009.

[75] David C Lee, Martial Hebert, and Takeo Kanade. Geometric Reasoning for Single Image Structure Recovery. *Cvpr*, 2009.

[76] D.C. Lee, Abhinav Gupta, Martial Hebert, and Takeo Kanade. Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces. *Proc. NIPS*, 1:1–9, 2010.

[77] Luca Del Pero, Joshua Bowdish, Daniel Fried, Bonnie Kermgard, Emily Hartley, and Kobus Barnard. Bayesian geometric modeling of indoor scenes. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pages 2719–2726, 2012.

[78] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering free space of indoor scenes from a single image. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pages 2807–2814, 2012.

[79] Alexander G. Schwing and Raquel Urtasun. Efficient Exact Inference for 3D Indoor Scene Understanding. In *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, volume 7577 LNCS, pages 299–313. 2012.

[80] Axel Furlan, Stephen D. Miller, Domenico G. Sorrenti, Li Fei-Fei, and Silvio Savarese. Free your Camera: 3D Indoor Scene Understanding from Arbitrary Camera Motion. *Procedings Br. Mach. Vis. Conf. 2013*, pages 24.1–24.11, 2013.

[81] Sid Yingze Bao, Axel Furlan, Li Fei-Fei, and Silvio Savarese. Understanding the 3D layout of a cluttered room from multiple images. *2014 IEEE Winter Conf. Appl. Comput. Vision, WACV 2014*, pages 690–697, 2014.

[82] Jana Kosecka and Wei Zhang. Video Compass. *Eccv*, 2353:476–490, 2002.

[83] Alexander G Schwing, Tamir Hazan, Marc Pollefeys, and Raquel Urtasun. Efficient Structured Prediction for 3D Indoor Scene Understanding.

[84] Huayan Wang, Stephen Gould, and Daphne Koller. Discriminative learning with latent variables for cluttered indoor scene understanding. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 6314 LNCS(PART 4):497–510, 2010.

[85] Grace Tsai, Changhai Xu, Jingen Liu, and Benjamin Kuipers. Real-time indoor scene understanding using Bayesian filtering with motion cues. *Proc. IEEE Int. Conf. Comput. Vis.*, pages 121–128, 2011.

[86] Grace Tsai and Benjamin Kuipers. Toward visual semantic modeling of the local environment for an indoor navigating robot. 2012.

[87] Carsten Rother. A new approach to vanishing point detection in architectural environments. *Image Vis. Comput.*, 20(9-10):647–655, 2002.

[88] Olga Barinova, Victor Lempitsky, Elena Tretiak, and Pushmeet Kohli. Geometric Image Parsing in Man-Made Environments. Number 228180, pages 57–70. 2010.

[89] O Grau. A scene analysis system for the generation of 3-D models. In *Proceedings. Int. Conf. Recent Adv. 3-D Digit. Imaging Model. (Cat. No.97TB100134)*, pages 221–228. IEEE Comput. Soc. Press, 1997.

[90] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric context from a single image. *Proc. IEEE Int. Conf. Comput. Vis.*, I:654–661, 2005.

[91] Derek Hoiem, Alexei a. Efros, and Martial Hebert. Putting Objects in Perspective. *Int. J. Comput. Vis.*, 80(1):3–15, apr 2008.

[92] Abhinav Gupta, Alexei a. Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 6314 LNCS(PART 4):482–496, 2010.

[93] Lawrence GIlman Roberts. Machine perception of three-dimensional solids, 1965.

[94] Sid Yingze Bao, Min Sun, and Silvio Savarese. Toward coherent object detection and scene layout understanding. *Image Vis. Comput.*, 29(9):569–579, 2011.

[95] Sunando Sengupta, Paul Sturgess, L'ubor Ladicky, and Philip H. S. Torr. Automatic dense visual semantic mapping from street-level imagery. *2012 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 857–862, oct 2012.

[96] Uwe Franke and David Pfeiffer. The Stixel World - A Compact Medium Level Representation of the 3D-World. pages 1–10, 2009.

[97] David Pfeiffer and Uwe Franke. Efficient representation of traffic scenes by means of dynamic stixels. *IEEE Intell. Veh. Symp. Proc.*, pages 217–224, 2010.

[98] David Pfeiffer and Uwe Franke. Towards a Global Optimal Multi-Layer Stixel Representation of Dense 3D Data. *Bmvc '11*, pages 51.1–51.12, 2011.

[99] Gabriel J Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and Recognition Using Structure from

Motion Point Clouds. In *Comput. Vis. – ECCV 2008*, volume 5302, pages 44–57. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[100] Shenlong Wang, Sanja Fidler, and Raquel Urtasun. Holistic 3D scene understanding from a single geo-tagged image. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 07-12-June:3964–3972, 2015.

[101] G. Floros and B. Leibe. Joint 2D-3D temporally consistent semantic segmentation of street scenes. In *2012 IEEE Conf. Comput. Vis. Pattern Recognit.*, volume 1, pages 2823–2830. IEEE, jun 2012.

[102] Andreas Geiger, Martin Lauer, and Raquel Urtasun. A generative model for 3D urban scene understanding from movable platforms. *Cvpr 2011*, pages 1945–1952, jun 2011.

[103] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3D Traffic Scene Understanding From Movable Platforms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(5):1012–1025, may 2014.

[104] Hongyi Zhang, Andreas Geiger, and Raquel Urtasun. Understanding High-Level Semantics by Modeling Traffic Patterns. *2013 IEEE Int. Conf. Comput. Vis.*, pages 3056–3063, dec 2013.

[105] Marcus a. Brubaker, Andreas Geiger, and Raquel Urtasun. Lost! Leveraging the Crowd for Probabilistic Visual Self-Localization. *2013 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3057–3064, jun 2013.

[106] Isaac Miller, Mark Campbell, and Dan Huttenlocher. Map-aided localization in sparse global positioning system environments using vision and particle filtering. *J. F. Robot.*, 28(5):619–643, 2011.

[107] Ignacio Parra, Miguel Angel Sotelo, David F. Llorca, C. Fernandez, A. Llamazares, N. Hernandez, and I. Garcia. Visual odometry and map fusion for GPS navigation assistance. *2011 IEEE Int. Symp. Ind. Electron.*, pages 832–837, jun 2011.

[108] Georgios Floros, Benito van der Zander, and Bastian Leibe. OpenStreetSLAM: Global vehicle localization using OpenStreetMaps.

2013 IEEE Int. Conf. Robot. Autom., pages 1054–1059, may 2013.

[109] Clement Fouque, Philippe Bonnifait, and David Betaille. Enhancement of global vehicle localization using navigable road maps and dead-reckoning. In *2008 IEEE/ION Position, Locat. Navig. Symp.*, pages 1286–1291. IEEE, 2008.

[110] P. Bonnifait, P. Bouron, P. Crubille, and D. Meizel. Data fusion of four ABS sensors and GPS for an enhanced localization of car-like vehicles. *Proc. 2001 ICRA. IEEE Int. Conf. Robot. Autom. (Cat. No.01CH37164)*, 2:1597–1602, 2001.

[111] P Merriaux, Y Dupuis, P Vasseur, and X Savatier. Wheel Odometry-based Car Localization and Tracking on Vectorial Map (Extended Abstract). pages 1890–1891, 2014.

[112] Dupuis Yohan, Pierre Merriaux, Pascal Vasseur, and Xavier Savatier. Vehicle Positioning in Road Networks without GPS. *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC*, 2015-Octob:1803–1809, 2015.

[113] Wenjie Lu, Emmanuel Seignez, and Roger Reynaud. Probabilistic Error Model for a Lane Marking Based Vehicle Localization Coupled to Open Source Maps. pages 360–365, 2014.

[114] Avdhut Joshi and Michael R James. Joint Probabilistic Modeling and Inference of Intersection Structure. pages 1072–1078, 2014.

[115] Aparna Taneja, Luca Ballan, and Marc Pollefeys. Never Get Lost Again: Vision Based Navigation using StreetView Images. In *people.inf.ethz.ch*, pages 1–16, 2014.

[116] Georges Baatz, Kevin Köser, David Chen, Radek Grzeszczuk, and Marc Pollefeys. Leveraging 3D City Models for Rotation Invariant Place-of-Interest Recognition. *Int. J. Comput. Vis.*, 96(3):315–334, feb 2012.

[117] M. Cummins and P. Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *Int. J. Rob. Res.*, 27(6):647–665, jun 2008.

[118] Roberto Arroyo, Pablo F Alcantarilla, Luis M Bergasa, J Javier Yebes, and Sebastian Bronte. Fast and effective visual place

recognition using binary codes and disparity information. In *2014 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, number Iros, pages 3089–3094. IEEE, sep 2014.

[119] C. Fernandez, D. F. Llorca, C. Stiller, and M. A. Sotelo. Curvature-based curb detection method in urban environments using stereo and laser. In *2015 IEEE Intell. Veh. Symp.*, number Iv, pages 579–584. IEEE, jun 2015.

[120] Marvin Raaijmakers and Mohamed Essayed Bouzouraa. Circle detection in single-layer laser scans for roundabout perception. *17th Int. IEEE Conf. Intell. Transp. Syst.*, pages 2636–2643, 2014.

[121] Ruisheng Wang, Jeff Bach, and Frank P. Ferrie. Window detection from mobile LiDAR data. In *2011 IEEE Work. Appl. Comput. Vis.*, pages 58–65. IEEE, jan 2011.

[122] Olga Vysotska. Exploiting Building Information from Publicly Available Maps in Graph-Based SLAM. *Iros*, pages 1–6, 2016.

[123] Carlos Fernandez, Rubén Izquierdo Gonzalo, Fernandez Llorca David, and Miguel Angel Sotelo. A Comparative Analysis of Decision Trees Based Classifiers for Road Detection in Urban Environments. *18th IEEE Intell. Transp. Syst. Conf.*, pages 719–724, 2015.

[124] Britta Hummel, Werner Thiemann, and Irina Lulcheva. *Scene Understanding of Urban Road Intersections with Description Logic*. 2008.

[125] Francesco Visin, Kyle Kastner, Kyunghyun Cho, Matteo Matteucci, Aaron C. Courville, and Yoshua Bengio. Renet: A recurrent neural network based alternative to convolutional networks. *CoRR*, abs/1505.00393, 2015.

[126] Fisher Yu and Vladlen Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. *Iclr*, pages 1–9, nov 2015.

[127] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. 2016.

[128] Jason J. Corso. Discriminative modeling by Boosting on Multilevel Aggregates. In *2008 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1–8. IEEE, jun 2008.

[129] Z Tao, Ph Bonnifait, V. Fremont, and J. Ibanez-Guzman. Mapping and localization using GPS, lane markings and proprioceptive sensors. In *IEEE Int. Conf. Intell. Robot. Syst.*, 2013.

[130] Markus Schreiber, Carsten Knoppel, and Uwe Franke. LaneLoc: Lane marking based localization using highly accurate maps. In *2013 IEEE Intell. Veh. Symp.*, number Iv, pages 449–454. IEEE, jun 2013.

[131] B. Douillard, D. Fox, F. Ramos, and H. Durrant-Whyte. Classification and Semantic Mapping of Urban Environments. *Int. J. Rob. Res.*, 30(1):5–32, jul 2010.

[132] Philip David. Detecting Planar Surfaces in Outdoor Urban Environments. Technical report, DTIC Document, 2008.

[133] O Grau. 3-D Modelling of Buildings using High-Level Knowledge. *Proc. Comput. Graph. Int. 1998*, 1998.

[134] Marian Himstedt, Alen Alempijevic, Liang Zhao, Shoudong Huang, and Hans Joachim Boehme. Towards robust vision-based self-localization of vehicles in dense urban environments. *IEEE Int. Conf. Intell. Robot. Syst.*, pages 3152–3157, 2012.

[135] Bahman Soheilian, Olivier Tournaire, Nicolas Paparoditis, Bruno Vallet, and Jean-Pierre Papelard. Generation of an integrated 3D city model with visual landmarks for autonomous navigation in dense urban areas. In *2013 IEEE Intell. Veh. Symp.*, number Iv, pages 304–309. IEEE, jun 2013.

[136] Karl Ni, Nicholas Armstrong-Crews, and Scott Sawyer. Geo-registering 3D point clouds to 2D maps with scan matching and the Hough Transform. In Intergovernmental Panel on Climate Change, editor, *2013 IEEE Int. Conf. Acoust. Speech Signal Process.*, volume 53, pages 1864–1868, Cambridge, may 2013. IEEE.

[137] Jason T. Isaacs, Andrew T. Irish, Francois Quitin, Upamanyu Madhow, and Joao P. Hespanha. Bayesian localization and mapping using GNSS SNR measurements. In *2014 IEEE/ION Position, Locat. Navig. Symp. - PLANS 2014*, pages 445–451. IEEE, may 2014.

[138] Oliver Wulf, K.O. Arras, H.I. Christensen, and Bernardo Wagner. 2D mapping of cluttered indoor environments by means of 3D

perception. In *IEEE Int. Conf. Robot. Autom. 2004. Proceedings. ICRA '04. 2004*, number April, pages 4204–4209 Vol.4. IEEE, 2004.

[139] Kohei Ito, Keisuke Yokota, and Akihisa Ohya. Localization of mobile robot by matching three-dimensional data of SOKUIKI sensor and aerial imagery. *2012 IEEE/SICE Int. Symp. Syst. Integr.*, (2):49–54, dec 2012.

[140] Todd R. Kushner and Sunil Puri. Progress In Road Intersection Detection For Autonomous Vehicle Navigation. In Wendell H. Chun and William J. Wolfe, editors, *Proc. SPIE*, volume 0852, pages 19–24, jan 1987.

[141] Yihuan Zhang, Jun Wang, Xiaonian Wang, Chaocheng Li, and Liang Wang. 3D LIDAR-Based Intersection Recognition and Road Boundary Detection Method for Unmanned Ground Vehicle. In *2015 IEEE 18th Int. Conf. Intell. Transp. Syst.*, volume 2015-Octob, pages 499–504. IEEE, sep 2015.

[142] F Heimes, K Fleischer, and H.-H. Nagel. Automatic generation of intersection models from digital maps for vision-based driving on inner city intersections. *Intell. Veh. Symp. 2000. IV 2000. Proc. IEEE*, (Mi):498–503, 2000.

[143] G. Struck, J. Geisler, F Laubenstein, H.-H. Nagel, and G Siegle. Interaction Between Digital Road Map Systems And Trinocular Autonomous Driving. In *Proc. Intell. Veh. '93 Symp.*, pages 461–466. IEEE.

[144] W. Enkelmann, G. Struck, and J. Geisler. ROMA - a system for model-based analysis of road markings. *Proc. Intell. Veh. '95. Symp.*, pages 356–360, 1995.

[145] José M. Alvarez, Theo Gevers, Ferran Diego, and Antonio M. Lopez. Road Geometry Classification by Adaptive Shape Models. *IEEE Trans. Intell. Transp. Syst.*, 14(1):459–468, mar 2013.

[146] Ian Baldwin and Paul Newman. Laser-only road-vehicle localization with dual 2D push-broom LIDARS and 3D priors. In *2012 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2490–2497. IEEE, oct 2012.

[147] Wende Zhang. LIDAR-based road and road-edge detection. *IEEE Intell. Veh. Symp. Proc.*, pages 845–848, 2010.

[148] Wentao Yao, Zhidong Deng, and Lipu Zhou. Road curb detection using 3D lidar and integral laser points for intelligent vehicles. *6th Int. Conf. Soft Comput. Intell. Syst. 13th Int. Symp. Adv. Intell. Syst. SCIS/ISIS 2012*, pages 100–105, 2012.

[149] Gangqiang Zhao and Junsong Yuan. Curb detection and tracking using 3D-LIDAR scanner. *Proc. - Int. Conf. Image Process. ICIP*, pages 437–440, 2012.

[150] Tan Li and Deng Zhidong. A new 3D LIDAR-based lane markings recognition approach. In *2013 IEEE Int. Conf. Robot. Biomimetics*, number December, pages 2197–2202. IEEE, dec 2013.

[151] Danilo Habermann, Alberto Hata, Denis Wolf, and Fernando S. Osorio. 3D point clouds segmentation for autonomous ground vehicle. *Brazilian Symp. Comput. Syst. Eng. SBESC*, pages 143–148, 2014.

[152] R. Fernandes, C. Premebida, P. Peixoto, D. Wolf, and U. Nunes. Road Detection Using High Resolution LIDAR. *2014 IEEE Veh. Power Propuls. Conf.*, pages 1–6, 2014.

[153] Tongtong Chen, Bin Dai, Daxue Liu, Jinze Song, and Zhao Liu. Velodyne-based curb detection up to 50 meters away. *IEEE Intell. Veh. Symp. Proc.*, 2015-Augus(Iv):241–248, 2015.

[154] Ping Kuang, Qinxing Zhu, and Xudong Chen. A Road Lane Recognition Algorithm Based on Color Features in AGV Vision Systems. In *2006 Int. Conf. Commun. Circuits Syst.*, volume 1, pages 475–479. IEEE, jun 2006.

[155] Serge Beucher and Michel Bilodeau. Road Segmentation and Obstacle Detection by a Fast Water shed Transformat ion. *Proc. 1994 Intell. Veh. Symp.*, pages 296–301, 1994.

[156] Jinyou Zhang and H.-H. Nagel. Texture-based segmentation of road images. In *Proc. Intell. Veh. '94 Symp.*, pages 260–265. IEEE, 1994.

[157] Alberto Broggi. Robust real-time lane and road detection in critical shadow conditions. In *Proc. Int. Symp. Comput. Vis. - ISCV*, pages 353–358. IEEE Comput. Soc. Press, 1995.

[158] Keyu Lu, Jian Li, Xiangjing An, and Hangen He. A hierarchical approach for road detection. In *2014 IEEE Int. Conf. Robot. Autom.*, pages 517–522. IEEE, may 2014.

[159] David M. Sarver and Mark Yanosky. Principles of cosmetic dentistry in orthodontics: Part 3. Laser treatments for tooth eruption and soft tissue problems. *Am. J. Orthod. Dentofac. Orthop.*, 127(2):262–264, feb 2005.

[160] Huan Wang and Yan Gong. Road Detection via Superpixels and Interactive Image Segmentation. *4th Annu. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst.*, pages 152–155, 2014.

[161] José M. Álvarez Alvarez and Antonio M. Lopez. Road Detection Based on Illuminant Invariance. *IEEE Trans. Intell. Transp. Syst.*, 12(1):184–193, mar 2011.

[162] Y. He, H. Wang, and B. Zhang. Color-Based Road Detection in Urban Traffic Scenes. *IEEE Trans. Intell. Transp. Syst.*, 5(4):309–318, 2004.

[163] Jose M. Alvarez, Theo Gevers, and Antonio M. Lopez. Vision-based road detection using road models. In *2009 16th IEEE Int. Conf. Image Process.*, pages 2073–2076. IEEE, nov 2009.

[164] Zhen He, Tao Wu, Zhipeng Xiao, and Hangen He. Robust road detection from a single image using road shape prior. *2013 IEEE Int. Conf. Image Process. ICIP 2013 - Proc.*, 0:2757–2761, 2013.

[165] Miguel Angel Sotelo, Francisco Javier Rodriguez, and Luis Magdalena. VIRTUOUS: Vision-based road transportation for unmanned operation on urban-like scenarios. *IEEE Trans. Intell. Transp. Syst.*, 5(2):69–83, 2004.

[166] J.B. Mena. State of the art on automatic road extraction for GIS update: a novel classification. *Pattern Recognit. Lett.*, 24(16):3037–3058, dec 2003.

[167] Fa-mao YE, Lin SU, and Jiang-long TANG. Automatic Road Extraction Using Particle Filters from High Resolution Images. *J. China Univ. Min. Technol.*, 16(4):490–493, dec 2006.

[168] Shenlong Wang, Sanja Fidler, and Raquel Urtasun. HD Maps : Fine-grained Road Segmentation by Parsing Ground and Aerial

Images. *2016 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3611–3619, 2016.

[169] Florian Kuhnt, J Marius Zöllner, Florian Kuhnt, Stefan Orf, Sebastian Klemm, and J Z Marius. Lane-precise Localization of Intelligent Vehicles Using the Surrounding Object Constellation Lane-precise Localization of Intelligent Vehicles Using the Surrounding Object Constellation. (November), 2016.

[170] Tobias Kuhnl, Franz Kummert, and Jannik Fritsch. Visual ego-vehicle lane assignment using Spatial Ray features. *IEEE Intell. Veh. Symp. Proc.*, (Iv):1101–1106, 2013.

[171] Johannes Rabe, Marc Necker, and Christoph Stiller. Ego-lane estimation for lane-level navigation in urban scenarios. In *2016 IEEE Intell. Veh. Symp.*, volume 2016-Augus, pages 896–901. IEEE, jun 2016.

[172] Soomok Lee, Seong Woo Kim, and Seung Woo Seo. Accurate ego-lane recognition utilizing multiple road characteristics in a Bayesian network framework. *IEEE Intell. Veh. Symp. Proc.*, 2015-Augus(Iv):543–548, 2015.

[173] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *J. Basic Eng.*, 82(1):35, 1960.

[174] J.C. McCall and M.M. Trivedi. Video-Based Lane Estimation and Tracking for Driver Assistance: Survey, System, and Evaluation. *IEEE Trans. Intell. Transp. Syst.*, 7(1):20–37, mar 2006.

[175] Sibel Yenikaya, Gökhan Yenikaya, and Ekrem Düven. Keeping the vehicle on the road. *ACM Comput. Surv.*, 46(1):1–43, oct 2013.

[176] Augusto Luis Ballardini, Daniele Cattaneo, Simone Fontana, and Domenico Giorgio Sorrenti. Leveraging the OSM building data to enhance the localization of an urban vehicle. In *2016 IEEE 19th Int. Conf. Intell. Transp. Syst.*, pages 622–628. IEEE, nov 2016.

[177] Andreas Geiger, Julius Ziegler, and Christoph Stiller. StereoScan: Dense 3d reconstruction in real-time. *2011 IEEE Intell. Veh. Symp.*, (Iv):963–968, jun 2011.

[178] OpenStreetMap. ©OpenStreetMap contributors.

[179] Google. Google Maps Additional Terms of Service 2(f), 2012.

[180] Osmium. http://osmcode.org/osmium.

[181] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, 2008.

[182] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *2012 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2684–2689. IEEE, oct 2012.

[183] Konstantinos G Derpanis. Overview of the ransac algorithm. 2010.

[184] Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[185] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient Large-Scale Stereo Matching. pages 25–38. 2011.

[186] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[187] Anna Petrovskaya and Sebastian Thrun. Model based vehicle detection and tracking for autonomous urban driving. *Auton. Robots*, 26(2-3):123–139, 2009.

[188] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *2011 IEEE Int. Conf. Robot. Autom.*, pages 1–4. IEEE, may 2011.

[189] Carlos Fernández López. *Road Scene Interpretation For Autonomous Navigation Fusing Stereo Vision and Digital Maps*. PhD thesis, 2016.

[190] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. *Comput. Vision–ECCV 2006*, pages 1–15, 2006.

[191] Thomas Leung and Jitendra Malik. No Title. *Int. J. Comput. Vis.*, 43(1):29–44, 2001.

[192] Vibhav Vineet, Ondrej Miksik, Morten Lidegaard, Matthias Niebner, Stuart Golodetz, Victor A Prisacariu, Olaf Kahler, David W Murray, Shahram Izadi, Patrick Peerez, and Philip H S Torr. Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In *2015 IEEE Int. Conf. Robot. Autom.*, pages 75–82. IEEE, may 2015.

[193] Giovani Bernardes Vitor, Alessandro C Victorino, Janito V Ferreira, Giovani Bernardes Vitor, Alessandro C Victorino, Janito V Ferreira A, Giovani B Vitor, Alessandro C Victorino, and Janito V Ferreira. A probabilistic distribution approach for the classification of urban roads in complex environments To cite this version : A probabilistic distribution approach for the classification of urban roads in complex environments. 2014.

[194] Philipp Krähenbühl and Vladlen Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *Nips'11*, (4):1–9, oct 2012.

[195] Andreas C. Müller. Superpixel based semantic segmentation. https://github.com/amueller/segmentation, 2013.

[196] Andrea Vedaldi and Stefano Soatto. Quick Shift and Kernel Methods for Mode Seeking. In *Comput. Vis. – ECCV 2008*, pages 705–718. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[197] R Achanta, A Shaji, K Smith, A Lucchi, P Fua, and Sabine Susstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2274–2282, nov 2012.

[198] Jose M Alvarez, Theo Gevers, Yann Lecun, and Antonio M Lopez. LNCS 7578 - Road Scene Segmentation from a Single Image. pages 376–389, 2012.

[199] F Smarandache and J Dezert. *Advances and Applications of DSmT for Information Fusion (Collected works), second volume: Collected Works*, volume 1. 2015.

[200] Quanwen Zhu, Long Chen, Qingquan Li, Ming Li, Andreas Nuchter, and Jian Wang. 3D LIDAR point cloud based intersection recognition for autonomous driving. In *2012 IEEE Intell. Veh. Symp.*, pages 456–461. IEEE, jun 2012.

[201] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pages 3354–3361, 2012.

[202] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV)*, 2011.

[203] Ac Müller and Sven Behnke. PyStruct-Learning Structured Prediction in Python. *J. Mach. Learn. Res.*, 15:2055–2060, 2013.

[204] L'ubor Ladicky, Chris Russell, Pushmeet Kohli, and Philip H S Torr. Associative Hierarchical Random Fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(6):1056–1077, jun 2014.

[205] Andrea Vedaldi and Stefano Soatto. *Computer Vision – ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV*, chapter Quick Shift and Kernel Methods for Mode Seeking, pages 705–718. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[206] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels. Technical report, 2010.

[207] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

Augusto Luis Ballardini
PhD Thesis
27th March, 2017