**UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS**

Luiz Ricardo Takeshi Horita

**Detecção de Características Visuais de Vias Urbanas
Usando Rede Neural Convolutiva com Aplicação em
Localização de Veículo**

São Carlos

2018

Luiz Ricardo Takeshi Horita

**Visual Urban Road Features Detection Using
Convolutional Neural Network with Application on
Vehicle Localization**

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, para obtenção do título de Mestre em Ciências - Programa de Pós-Graduação em Engenharia Elétrica.

Área de concentração: Sistemas Dinâmicos

Supervisor: Prof. Dr. Valdir Grassi Junior

São Carlos

2018

Trata-se da versão corrigida da dissertação. A versão original se encontra disponível na EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica.

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

H811d Horita, Luiz Ricardo Takeshi
 Detecção de características visuais de vias urbanas usando rede neural convolutiva com aplicação em localização de veículo / Luiz Ricardo Takeshi Horita; orientador Valdir Grassi Junior. São Carlos, 2018.

 Dissertação (Mestrado) - Programa de Pós-Graduação em Engenharia Elétrica e Área de Concentração em Sistemas Dinâmicos -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2018.

 1. Detecção de guia. 2. Detecção de sinalização horizontal. 3. Aprendizado de máquina. 4. Rede neural convolutiva. 5. Visão estéreo. 6. Localização de veículos. I. Título.

FOLHA DE JULGAMENTO

Candidato: Engenheiro **LUIZ RICARDO TAKESHI HORITA**.

Título da dissertação: "Detecção de características visuais de vias urbanas usando rede neural convolutiva com aplicação em localização de veículo".

Data da defesa: 28/02/2018.

Comissão Julgadora:

Resultado:

Prof. Doutor **Valdir Grassi Junior**
(Orientador)
(Escola de Engenharia de São Carlos/EESC)

Aprovado

Prof. Dr. **Fernando Santos Osório**
(Instituto de Ciências Matemáticas e de Computação/ICMC)

APROVADO

Prof. Dr. **Douglas Guimarães Macharet**
(Universidade Federal de Minas Gerais/UFMG)

Aprovado

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica:
Prof. Associado **Luis Fernando Costa Alberto**

Presidente da Comissão de Pós-Graduação:
Prof. Associado **Luis Fernando Costa Alberto**

I dedicate this work to my family. Specially to my parents, Ricardo Yoshio Horita and Marlei Shizue Takizawa Horita.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Prof. Valdir Grassi Junior for giving orientation and feedback along the last two years.

I am also grateful to everyone that somehow helped me during all the process of my project development. The assistance and the material support given by the Mobile Robotics Laboratory (LRM) were essential to execute experiments. Among the people who helped me, I would like to acknowledge principally my girlfriend Angelica Tiemi Mizuno Nakamura, who has been there whenever I needed; and Luis Alberto Rosero for having immensely helped me during data collection and tests with the CaRINA II platform.

Thanks to Foundation for Engineering Technological Development (FDTE) for financing my Master's project development during the last year.

Last but not the least, to the Intelligent Systems Laboratory (LASI) members, thanks for making this journey pleasant with your friendship. For my family that has always been present and encouraging me to go further, thank you very much. Finally, an unmeasurable gratitude for my parents, who have never measured effort to give me all support for achieving my goals.

ABSTRACT

HORITA, L. R. T. **Visual Urban Road Features Detection Using Convolutional Neural Network with Application on Vehicle Localization.** 2018. 105p. Dissertation (Masters in Science) - São Carlos School of Engineering, University of São Paulo, São Carlos, 2018.

Curbs and road markings were designed to provide a visual low-level spatial perception of road environments. In this sense, a perception system capable of detecting those road features is of utmost importance for an autonomous vehicle. In vision-based approaches, few works have been developed for curb detection, and most of the advances on road marking detection have aimed lane markings only. Therefore, to detect all these road features, multiple algorithms running simultaneously would be necessary. Alternatively, as the main contribution of this work, it was proposed to employ an architecture of Fully Convolutional Neural Network (FCNN), denominated as 3CSeg-Multinet, to detect curbs and road markings in a single inference. Since there was no labeled dataset available for training and validation, a new one was generated with Brazilian urban scenes, and they were manually labeled. By visually analyzing experimental results, the proposed approach has shown to be effective and robust against most of the clutter present on images, running at around 10 fps in a Graphics Processing Unit (GPU). Moreover, with the intention of granting spatial perception, stereo vision techniques were used to project the detected road features in a point cloud. Finally, as a way to validate the applicability of the proposed perception system on a vehicle, it was also introduced a vision-based metric localization model for the urban scenario. In an experiment, compared to the ground truth, this localization method has revealed consistency on its pose estimations in a map generated by LIDAR.

Keywords: Curb detection. Road marking detection. Machine learning. Convolutional neural network. Stereo Vision. Vehicle localization.

RESUMO

HORITA, L. R. T. **Detecção de Características Visuais de Vias Urbanas Usando Rede Neural Convolutiva com Aplicação em Localização de Veículo.** 2018. 105p. Dissertação (Mestrado em Ciências) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

Guias e sinalizações horizontais foram projetados para fornecer a percepção visual de baixo nível do espaço das vias urbanas. Deste modo, seria de extrema importância para um veículo autônomo ter um sistema de percepção capaz de detectar tais características visuais. Em abordagens baseadas em visão, poucos trabalhos foram desenvolvidos para detecção de guias, e a maioria dos avanços em detecção de sinalizações horizontais foi focada na detecção de faixas apenas. Portanto, para que fosse possível detectar todas essas características visuais, seria necessário executar diversos algoritmos simultaneamente. Alternativamente, como sendo a principal contribuição deste trabalho, foi proposto a adoção de uma Rede Neural Totalmente Convolutiva, denominado 3CSeg-Multinet, para detectar guias e sinalizações horizontais em apenas uma inferência. Como não havia um conjunto de dados rotulados disponível para treinar e validar a rede, foi gerado um novo conjunto com imagens capturadas em ambiente urbano brasileiro, e foi realizado a rotulação manual. Através de uma análise visual dos resultados experimentais obtidos, o método proposto mostrou-se eficaz e robusto contra a maioria dos fatores que causam confusão nas imagens, executando a aproximadamente 10 *fps* em uma GPU. Ainda, com o intuito de garantir a percepção espacial, foram usados métodos de visão estéreo para projetar as características detectadas em nuvem de pontos. Finalmente, foi apresentado também um modelo de localização métrica baseado em visão para validar a aplicabilidade do sistema de percepção proposto em um veículo. Em um experimento, este método de localização revelou-se capaz de manter as estimativas consistentes com a verdadeira pose do veículo em um mapa gerado a partir de um sensor LIDAR.

Palavras-chave: Detecção de guia. Detecção de sinalização horizontal. Aprendizado de máquina. Rede neural convolutiva. Visão estéreo. Localização de veículos.

LIST OF FIGURES

Figure 1 – Illustration of arbitrary MLP and CNN architectures.	28
Figure 2 – Illustration of arbitrary typical CNN and an encoder-decoder FCNN architectures.	29
Figure 3 – Illustration of a convolutional layer.	30
Figure 4 – Illustration of max-pooling.	31
Figure 5 – Illustration of a deconvolution with stride $s_{deconv} = 2$ and a kernel of 4×4	32
Figure 6 – The pinhole camera model.	33
Figure 7 – The perspective camera model.	34
Figure 8 – The perspective camera model with the camera’s parameters.	35
Figure 9 – Visual example of correlation-based matching. The blue correlation window is associated with a pixel $\mathbf{p}_1$ from I_1 , and its equivalent position in I_2 is illustrated in dashed lines. In image I_2 , the algorithm tries to find a matching in the search region represented in yellow. When it is found (in green), the disparity is given by the displacement between the positions $\mathbf{p}_1$ and $\mathbf{p}_2$	38
Figure 10 – Illustration of the epipolar geometry of an arbitrary stereo system. . . .	39
Figure 11 – Rectification of a stereo pair.	40
Figure 12 – 3D reconstruction by triangulation with nonintersecting rays.	42
Figure 13 – Illustration of the perception model. For simplicity, the illustration considers that the z axis from every reference frames are parallels. . . .	44
Figure 14 – (a) Example of a random environment, and (b) its Likelihood Field representation. The brighter a location, the higher is the likelihood value. . . .	45
Figure 15 – Example of two consecutive iterations of MCL.	47
Figure 16 – The four main modules for features detection.	49
Figure 17 – Examples of approaches used to reduce the illumination effects on images. The first column shows the original images; the second shows the chroma map proposed by Katramados, Crumpler and Breckon (2009); and the third shows the hue channel of HSI representation.	50
Figure 18 – Examples of lane markings detection performances in case of using (a) top-hat filter and (b) gray-level ridges.	51
Figure 19 – Curb detection according to its step-shape.	52
Figure 20 – MultiNet performance on road estimation. The green plot is the resulting estimation.	55
Figure 21 – SegNet performance on urban scenes segmentation for day and dusk test samples.	55

Figure 22 – The five factors to be considered on a metric localization approach. . .	60
Figure 23 – Description of Markov localization in diagram.	63
Figure 24 – Diagram with detailed network architecture. The input RGB image can be of any <i>size</i> , and the output is given by a two channel array of the same <i>size</i>	66
Figure 25 – Illustration of the steps for road features projection to a point cloud. .	67
Figure 26 – Diagram of the proposed localization model.	69
Figure 27 – Point cloud to laser scan conversion model.	70
Figure 28 – An example of conversion from point cloud to laser scan measurement.	71
Figure 29 – A sample from CaRINA dataset (SHINZATO et al., 2016) and from additional collected images. (a and c) Original images with ROI defined in red, (b and d) ground truth.	75
Figure 30 – Convergence of the first training process.	76
Figure 31 – Images that the FCNN fails to detect road features correctly.	77
Figure 32 – Convergence of the second training process.	77
Figure 33 – Random images used for test. The left column of images shows the results obtained after the first training, while the right column shows the results obtained after the second training.	79
Figure 34 – Stereo camera and LIDAR installation on CaRINA II platform.	80
Figure 35 – Calibration between sensors (<i>Velodyne</i> and <i>Bumblebee</i>) and the vehicle's frame (<i>base_link</i>) displayed in RVIZ. In this illustration, the vehicle's frame is denominated <i>base_link</i> , and <i>base_footprint</i> is the coordinate system of the vehicle in ground level. The yellow arrows represent the translation and rotation transformations between two coordinate systems.	81
Figure 36 – Comparison between point cloud given by LIDAR (colorful 32 concentric rings of points) and the road features point cloud (in white).	82
Figure 37 – Some good results for road features detection and projection. On images, the detected road markings are highlighted in green, while curbs are in blue.	83
Figure 38 – Some bad results for road features detection and projection. On images, the detected road markings are highlighted in green, while curbs are in blue.	84
Figure 39 – Road features projection tilted because of a speed bump.	84
Figure 40 – Maps of the place where localization experiment was executed.	85
Figure 41 – Example of the point cloud to laser scan conversion in case of speed bump. Point clouds are represented in green, and measured points by laser scan in red.	86
Figure 42 – Some results of the point cloud to laser scan conversion. Point clouds are represented in green, and measured points by laser scan in red. . .	87

Figure 43 – Examples of longitudinal imprecision during localization. Particles are represented in small red arrows, the estimated pose is in green, and the ground truth in blue. 89

LIST OF TABLES

Table 1 – A summary of the correlation-based correspondence (or block matching) algorithm.	38
Table 2 – Monte Carlo Localization, an algorithm based on the particle filters. . .	46
Table 3 – Existing visual approaches for metric localization.	58
Table 4 – Mean errors for each region of the map.	88
Table 5 – Absolute average errors for each region of the map.	88

LIST OF ABBREVIATIONS AND ACRONYMS

ANN	Artificial Neural Network
BM	Block Matching
BRIEF	Binary Robust Independent Elementary Features
CaRINA	Intelligent Robotic Car for Autonomous Navigation
CNN	Convolutional Neural Network
CRob-SC	Center for Robotics of University of São Paulo at São Carlos
EESC	São Carlos School of Engineering
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
FCNN	Fully Convolutional Neural Network
FFT	Fast Fourier Transform
FOP	Focus of Projection
GNSS	Global Navigation Satellite System
GPOM	Gaussian Process Occupancy Map
GPS	Global Positioning System
GPU	Graphics Processing Unit
HSI	Hue, Saturation and Intensity
ICMC	Institute of Mathematical and Computer Sciences
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INCT-SEC	Brazilian National Institute of Science and Technology for Critical Embedded System
KF	Kalman Filter
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute

LASI	Laboratory of Intelligent Systems
LIDAR	Light Detection And Ranging
LRM	Mobile Robotics Laboratory
LTS	Least Trimmed Squares
MCL	Monte Carlo Localization
MIT	Massachusetts Institute of Technology
MLP	Multilayer Perceptron
OGM	Occupancy Grid Map
OSM	Open Street Map
PCA	Principal Component Analysis
RANSAC	Random Sample Consensus
RGB	Red, Green and Blue
RNN	Recurrent Neural Network
ROI	Region Of Interest
RTK	Real Time Kinematic
SGBM	Semi-Global Block Matching
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SLT	Statistical Learning Theory
SONAR	Sound Navigation and Ranging
SURF	Speeded Up Robust Features
TN	True Negative
TP	True Positive
USP	University of São Paulo
USP-SC	Campus São Carlos of USP
VC-dimension	Vapnik-Chervonenkis dimension

CONTENTS

1	INTRODUCTION	23
2	BACKGROUND CONCEPTS	27
2.1	Convolutional Neural Network	27
2.1.1	Convolution Arithmetic	30
2.1.2	Pooling Arithmetic	31
2.1.3	Deconvolution Arithmetic	32
2.2	The Perspective Camera	33
2.2.0.1	Extrinsic Parameters	34
2.2.0.2	Intrinsic Parameters	35
2.3	The Stereo System	36
2.3.1	The Correspondence Problem	37
2.3.2	Epipolar Geometry	39
2.3.2.1	Rectification	40
2.3.2.2	Reconstruction	41
2.4	Occupancy Grid Maps	42
2.4.1	Likelihood Fields	43
2.5	Monte Carlo Localization	45
3	RELATED WORKS	49
3.1	Road Features Detection	49
3.1.1	A Survey	49
3.1.1.1	Image Pre-processing	49
3.1.1.2	Features Extraction	51
3.1.1.3	Model Fitting	53
3.1.1.4	Temporal Integration	53
3.1.2	Deep Learning for Road Scenes Segmentation	54
3.2	Visual Approaches for Metric Localization	56
3.3	Metric Localization	59
3.3.1	Map Features	60
3.3.2	Map Representation	61
3.3.3	Matching Approach	62
3.3.4	Probabilistic Filtering	62
3.3.5	Sensors	64
4	ROAD FEATURES DETECTION	65

4.1	The 3CSeg-Multinet Architecture	65
4.2	Features Projection to Point Cloud	67
5	THE LOCALIZATION MODEL	69
5.1	The Measurement Model	70
6	EXPERIMENTS AND RESULTS	73
6.1	Road Features Detection	73
6.1.1	Training Scheme	73
6.1.2	Experimental Framework	74
6.1.2.1	Dataset	74
6.1.2.2	Setup	75
6.1.3	Results	76
6.2	3D Projection of Road Features	80
6.2.1	Experimental Framework	80
6.2.2	Results	81
6.3	Vision-Based Localization	85
6.3.1	Experimental Framework	85
6.3.2	Results	86
7	CONCLUSION	91
7.1	Future Works	92
	BIBLIOGRAPHY	95
	APPENDIX	103
	APPENDIX A – PUBLICATION	105
A.1	Related Paper	105
A.2	Others	105

1 INTRODUCTION

In early times car was considered as luxurious, but it has become essential for daily life over the years, resulting in a fast growth of car fleet. But, simultaneously, traffic accidents rate has become larger. According to the World Health Organization (WHO), every year there are around 1.3 million fatal accidents, which correspond to the 10th main cause of death in the world¹. In Brazil, fatal traffic accidents represent the fifth main cause of death, amounting around 45 thousand life losses per year². Even though preventive measures are taken from governments (e.g. speed limits settings, traffic lights installations, alert signalizations, etc.), the mortality rate in roads are still high due to human failures and imprudences. For this reason, targeting traffic safety improvement, car companies and communities of robotics field researchers have been investing in Advanced Driver Assistance Systems (ADAS) for partial automation of the vehicles, and even totally autonomous vehicles (or self-driving cars).

In both cases, it is crucial to have robust perception systems that allow the vehicle to understand the environment where it is navigating. In this way, the vehicle may reliably assist the motorist, or even navigate by itself. Among all perception systems that may comprise an intelligent or an autonomous vehicle, the one responsible for recognizing curbs and road markings in the urban scenario plays a fundamental role. This is because these road features³ were designed to provide a low-level perception of the road environment, which can be exploited by Lane Departure Warning Systems (LDWS), Lane Keeping Systems (LKAS), and for Localization Systems (LU et al., 2014).

Recently, LIDARs and cameras have been widely employed for road features detection (ZIEGLER et al., 2014; HATA; WOLF, 2016). Because LIDARs (more specifically, multilayer LIDARs) provide straightforward and precise metric measurement, they are more frequently used for curb detection. But, it is also possible to detect road markings with this type of sensor by thresholding the reflective intensity of laser beams that point to the road surface. However, more robust road marking detection can be reached by visual approaches. The reason for this is that cameras provide a denser amount of data in form of images, from which is possible to get important information (e.g. color, texture, shape, and even depth in case of stereo images). Nevertheless, vision-based curb detection is more challenging.

¹ Source: WHO. **The top 10 causes of death**. 2017. Available from Internet: <<http://www.who.int/mediacentre/factsheets/fs310/en/>>. Accessed in: jan 2018.

² Source: CAMARGO, A.B.M. MAIA, P.B. **SP Demográfico: Resenha de Estatísticas Vitais do Estado de São Paulo**. São Paulo: Fundação Sistema Estadual de Análise de Dados, 2017.

³ Notation: In this document, the term *road features* is referred to curbs and road markings.

In vision-based approaches, the problem of road features detection consists in classifying each pixel of an image as road feature or non-road feature. However, usually different methods are applied to each type of feature (curb and road marking). Because curbs have a characteristic step shape, some authors proposed techniques based on depth estimation by stereo vision to detect small and abrupt surface elevation (ONIGA et al., 2007; PRADEEP; MEDIONI; WEILAND, 2008). But, considering that the depth resolution decreases according to the distance from the stereo camera, the accuracy of curb detection stays limited to very short distances.

In case of road markings, since they are drawn in bright colors over the dark road surface, most existing approaches rely on this high contrast pattern (DU; TAN, 2016a). However, in practice, cluttered scenes make road marking detection a challenging task, making it necessary to have additional techniques for false detections filtering. In the past few years, most advances on road marking detection have aimed lane markings only, due to applications on systems that require lane-level localization (HILLEL et al., 2014). Still, there were some works for detection of other road markings, but a different algorithm had to be used for each different pattern (DU; TAN, 2016a; VEIT et al., 2008; SEBSADJI et al., 2010).

Besides recent advances on methods for road features detection, there was also a significant progress in deep learning studies. Deep learning is not a new concept, but due to its high computational cost for training and hardware limitations, its applicability was prohibitive in early studies (SCHMIDHUBER, 2014). However, with a more efficient architecture based on Convolutional Neural Network (CNN) and a powerful Graphics Processing Unit (GPU), a deep learning approach has achieved the state-of-the-art performance for image classification (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Since then, its adoption has become very popular to solve several other computer vision tasks (DUMOULIN; VISIN, 2016). Among all applications that the deep learning has shown state-of-the-art performance, the most interesting for this work is the image segmentation in multiple classes, since the goal of road features detection is to make a pixel-wise classification. For instance, Fully Convolutional Neural Networks (FCNNs) in encoder-decoder architectures have been achieving the best performances for image segmentation (TEICHMANN et al., 2016; BADRINARAYANAN; KENDALL; CIPOLLA, 2017).

Therefore, as the main contribution, this work proposes to employ a deep FCNN for road features detection. In this way, it is not necessary to use multiple algorithms running in the same image to detect different road marking patterns, and a method based on metric information estimated by stereo vision to detect curbs. Instead, all of them can be detected in a single inference without the necessity of additional filtering techniques. The adopted architecture was an adaption from a part of the MultiNet network (TEICHMANN et al., 2016) responsible for image segmentation. The adapted architecture is denominated

in this work as 3Cseg-Multinet, because it can segment the image in three different classes (road markings, curbs and background). This contribution was published at the 14th Latin American Robotics Symposium, as shown in Appendix A.1.

For experiments, there was no labeled dataset available for road features detection benchmark, so a new one had to be manually generated. But, it was labeled only enough data to train and validate the network. Experimental results have shown that the adopted network can converge in a relative few training iterations if it is initialized with well-conditioned parameters. Moreover, in a visual analysis during tests, the proposed approach has shown efficiency and robustness against most clutter, running at around 10 frames per second (fps) in a GPU.

Road features detection on images can provide the vehicle an understanding of the scenario it is inserted in, but they are not enough to give a spatial perception of the environment. Thus, this work also proposes the use of stereo vision techniques for 3D projection of the detected road features. Basically, it uses the segmented image as a mask to filter a dense disparity map of the original image, outputting a segmented disparity map. Then, it applies a simple 3D reconstruction algorithm to project the detected road features in a point cloud.

Going further, aiming to verify the applicability of the proposed approach for road features detection and their projection in point clouds, this work comes up with the third and last proposal, which is a vision-based metric localization approach. The proposed localization method is based on Monte Carlo Localization (MCL) algorithm with Likelihood Fields, due to their robustness against ambiguous situations and imprecise sensor measurements (THRUN; BURGARD; FOX, 2006). But, since Likelihood Fields is actually a measurement model for range sensors, an adaption was applied to simulate a laser scan sensor over the point cloud. In summary, the measurement model of the proposed localization system initially detects road features from an image, project them to a point cloud by stereo vision algorithms, convert it to a laser scan measurement, and finally apply the Likelihood Fields algorithm to match the measurements with the map. The experiment was performed offline with data collected through the CaRINA II platform, and the map was kindly provided by Hata and Wolf (2016). Experimental results have shown a consistent localization performance that, although not enough to compete with the state-of-the-art, assure the applicability of the two first proposals.

The rest of this dissertation document is composed by the following chapters:

- Chapter 2: An explanation about the requisite background concepts for this work.
- Chapter 3: Review of the state of art on road features detection, and related works to this project.

- Chapter 4: A detailed description of the proposed approaches for road features detection and their 3D projection.
- Chapter 5: A detailed explanation of the proposed vision-based metric localization model.
- Chapter 6: Presentation of achieved results and discussion.
- Chapter 7: Conclusion and final considerations of this work.

2 BACKGROUND CONCEPTS

In this chapter some important concepts for this project are explained. Firstly, the CNN concept and its arithmetic are described. Then, to know how to get the depth information from images obtained by two cameras in different positions, it is given an overview about the perspective camera model and the stereo vision systems. The explanations of these topics were mostly based on (TRUCCO; VERRI, 1998). Finally, occupancy grid map, Likelihood Fields, and MCL are also explained, since a vision-based localization is explored as an application of the proposed road features detection in 3D space.

2.1 Convolutional Neural Network

Artificial Neural Networks (ANN) are computational models inspired on nervous systems of living beings, and are capable of acquiring and storing knowledge based on information. These models are defined by a set of processing units, which consists of *artificial neurons* connected with each other by *artificial synapses* represented by weights (SILVA; SPATTI; FLAUZINO, 2010). Since the late 80's, the most popular ANN was the *Multilayer Perceptron* (MLP) due to its capability of generating very complex functions for solving several problems, as pattern recognition and numerical regression.

In standard structure of a MLP, there is at least one hidden layer between the input and output layers. Each layer is formed by artificial neurons connected with all neurons from the adjacent layers by multiplier weights, as shown by Figure 1(a). But, depending on the problem a deep architecture (i.e. with many layers) is required to learn more complex features. Consequently, the computational cost and the Vapnik-Chervonenkis dimension increases significantly, which means that the network would spend much more time on training process, not to mention the reduction of its capability of generalization. Therefore, the standard MLP is practically unfeasible for this kind of situation.

To solve this, LeCun et al. (1989) applied the backpropagation algorithm to the *Neocognitron* network, which had been inspired by the visual cortex of cats in the work of Fukushima (1980). On this model, densely connected hidden layers were replaced by convolutional layers. Later, it was denominated as *Convolutional Neural Network* (CNN). The replacement of multiplier weights by convolutions enabled the network to reduce its complexity by decreasing the number of connections. Figure 1 shows that even when the MLP and CNN have the same number of neurons in two adjacent layers, the CNN has much less parameters than the MLP. This is because the weights in convolutional layers are shared for every neuron.

Figure 1 – Illustration of arbitrary MLP and CNN architectures.

As MLP, CNN is a multiple layers network, but it was designed to recognize patterns that can be represented by a grid-like topology (GOODFELLOW; BENGIO; COURVILLE, 2016), like images. Since it works with kernels of shared weights and convolution, it works like filters to extract relevant features, being invariant to scale, rotation, distortion or translation. On this type of network, each neuron from a convolutional layer receives values from a *receptive field* of the previous layer, and the set of many neurons in a planar structure form a *feature map*. Each convolutional layer consists of many stacked feature maps (Figure 3), allowing the network to extract several relevant features.

Moreover, like in MLP, nonlinear activation functions (e.g. sigmoid and softmax) are used in CNN, but they are applied at the end of each convolutional layer instead of at the end of the network. Applying a nonlinear function on each convolutional layer allows better convergence during training process (LECUN et al., 1989).

Rectified linear unit function (ReLU) became very popular after Krizhevsky, Sutskever and Hinton (2012) showed empirically that it provides a faster convergence. In this work, ReLU was chosen as the nonlinear activation function, being described as

$$f(x) = \max(0, x). \quad (2.1)$$

Nonetheless, a typical CNN is not formed by convolutional layers only. After the feature maps extraction by convolutions, it is common to apply *pooling* by downsampling each feature map, reducing its resolution. This operation summarizes the output of a neighboring neurons in a feature map, allowing better invariance to small translations of the input (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; GOODFELLOW; BENGIO;

COURVILLE, 2016) . Thereby, successive layers alternating between convolution and pooling creates a *bipyramidal effect*, i.e. it scales down the spatial size but increase the number of feature maps (HAYKIN, 2001). This way, the next layer is capable to extract wider and more complex features from the input data, providing better contextual information for classification tasks. Finally, in the end of a typical CNN architecture, one or more fully connected layer (like an MLP) is used to estimate the classification of an input data. An arbitrary architecture of a typical CNN is illustrated in Figure 2(a).

Figure 2 – Illustration of arbitrary typical CNN and an encoder-decoder FCNN architectures.

(a) A conventional CNN architecture

(b) An encoder-decoder FCNN architecture

Source: author.

Although the bipyramidal effect allows to extract better contextual information of a wide region from the input data, it is not good for some applications like pixelwise classification, which is the case of this project. Since the bipyramidal effect reduces the spatial size of the feature maps, it is difficult to have an output with the same size of the input data. A way to solve this problem is to use the encoder-decoder architecture, replacing the fully connected layers by convolutional layers and adding layers of *deconvolution* (or

transposed convolution) to upsample the feature maps back to input size (MENDES; FREMONT; WOLF, 2016; ZEILER et al., 2010), as shown in Figure 2(b). An advantage of adopting this kind of architecture is that, besides providing pixelwise classifications, it is a FCNN, which allows to have an input of any spatial size.

Now that the key ideas of CNNs were discussed, one can understand the formal expressions of convolution, pooling and deconvolution.

2.1.1 Convolution Arithmetic

Assuming a tridimensional array L_k representing all stacked feature maps of the k -th layer and a kernel (or mask) K_1 containing the shared weights of a resulting feature map $L_{k+1}(i, j, 1)$, the convolution operation can be described by:

$$L_{k+1}(i, j, 1) = (L_k * K_1)(i, j) = \sum_m \sum_n \sum_c L_k(i - m, j - n, c) K_1(m, n, c). \quad (2.2)$$

Figure 3 shows a more intuitive way to understand the convolution, where it performs an element wise multiplication between the receptive field and kernel followed by the summation of all resulting elements.

Figure 3 – Illustration of a convolutional layer.

Source: author.

This operation is done on the whole feature maps L_k for each feature map of L_{k+1} , starting from the position $[1, 1]^T$ and moving on the horizontal and vertically with a stride s_{conv} in both directions. When the stride is unitary, one can perform convolution to produce a new feature map with the same spatial size of the previous layer by employing zero padding. This kind of padding is called *half padding* (or *same padding*).

Besides knowing how convolution works, it is also interesting to know the spatial relation between regions of a layer L_{k+1} and its previous layer L_k , i.e. how wide is the region in L_k from where a region in L_{k+1} extracts contextual information for classification. So, considering a square kernel of size w_k and stride s_{conv} , it is possible to find out the size w_i of the region in L_k from where a region of size w_o in L_{k+1} extracts contextual information by the following equation:

$$w_i = (w_o - 1)s_{conv} + w_k. \quad (2.3)$$

2.1.2 Pooling Arithmetic

For pooling, *max-pooling* has been largely employed (SIMONYAN; ZISSERMAN, 2014; TEICHMANN et al., 2016; BADRINARAYANAN; KENDALL; CIPOLLA, 2017). Basically, it takes the maximum value from a neighboring group of neurons in the same feature map, like shown in Figure 4. Like in convolutions described above, to produce a downsampled feature maps L_{k+1} , pooling operation is done on the whole feature maps L_k , moving with a stride s_{pool} . Mathematically, it is described by:

$$L_{k+1}(i, j) = \max_{N \times N} L_k(m, n), \quad (2.4)$$

where $N \times N$ defines the size of neighboring neurons.

Figure 4 – Illustration of max-pooling.

Source: author.

The equation that defines a relation between the size of a resulting region in L_{k+1} and the size of the region in L_k , from where the pooling operation extracts information, is given by the same one applied for convolution (Equation 2.3). But, instead stride s_{conv} , it is s_{pool} .

2.1.3 Deconvolution Arithmetic

Lastly, the deconvolution (or transposed convolution) computes a fractionally strided convolution, i.e., it spreads the pixels of feature maps in a wider space keeping a stride s_{deconv} between each pair of neighboring pixels and computes convolutions (Equation 2.2) with unitary strides ($s_{conv} = 1$) and half padding on these wider feature maps. So, to upsample a stack of feature maps by a factor of 2, one can apply a deconvolution of stride $s_{deconv} = 2$, as shown in Figure 5. The same applies for higher factors of upsampling.

Figure 5 – Illustration of a deconvolution with stride $s_{deconv} = 2$ and a kernel of 4×4 .

Source: author.

Since the spatial configuration of the input feature maps is changed in deconvolution, it is difficult to describe the relation between input and output sizes in a simple equation as done for convolution and pooling. This equation changes according to s_{deconv} and the convolution kernel size. For further details, one can consult the guide to convolution arithmetic elaborated by Dumoulin and Visin (2016).

Applying these concepts on a deep FCNN architecture, it may be possible to extract features, even in cluttered scenes. Assuming that a FCNN is already trained to detect curbs and road markings, it will output the image points that correspond to those features. The next step for the proposed localization approach is to estimate the 3D position of those features. For this, it is necessary to understand the concepts of the perspective camera and the stereo system, that are explained on the following two sections.

2.2 The Perspective Camera

The perspective camera model is a geometric model based on the pinhole model, that is illustrated in Figure 6. The *pinhole model* is an approximation of the mathematical relation between the 3D coordinate and the projection onto the image plane π . This approximation does not take the geometric distortion or blurring of unfocused objects into account. Also, it assumes an ideal pinhole, i.e. the hole is small enough so that all rays coming from outside of the dark box intersect in a unique point. Consequently, the projection is an inverted and focused image on the plane π .

Figure 6 – The pinhole camera model.

Source: author.

The *perspective camera model* is very similar to the pinhole model. The difference is that the image plane is reflected to the other side of the pinhole, keeping the same focal distance f from the hole and inverting the image to a plane π' as indicated on Figure 6. In this model, the pinhole $\mathbf{O}$ is denominated as the focus of projection (FOP) and π' as the image plane. The ${}^c z$ axis is called optical axis, and it intersects perpendicularly through π' on the principal point $\mathbf{o}'$ (or image center). Observing from Figure 7, it is possible to see that the image of $\mathbf{P}$, $\mathbf{p}$, is the point where the line $\mathbf{OP}$ intersects π' .

Considering the 3D coordinate of the camera frame in which $\mathbf{O}$ is the origin, and letting $\mathbf{P} = [{}^c X, {}^c Y, {}^c Z]^T$ and $\mathbf{p} = [{}^c x, {}^c y, f]^T$, it is possible to write the basic equations of perspective projections:

$$\begin{aligned} {}^c x &= f \frac{{}^c X}{{}^c Z} \\ {}^c y &= f \frac{{}^c Y}{{}^c Z} \end{aligned} \tag{2.5}$$

After addressing the perspective camera model, it is noticeable that the coordinates of a point in 3D space can be linked by equations with the coordinates of its corresponding image point. To do this, some camera's characteristics must be known. These characteristics are the *extrinsic* and *intrinsic* parameters.

Figure 7 – The perspective camera model.

Source: author.

2.2.0.1 Extrinsic Parameters

The extrinsic parameters are geometric parameters that defines the position and orientation of the camera frame with respect to a known reference frame, namely world reference frame. These parameters are illustrated in Figure 8 as orange arrows. Typically, this relation between both reference frames is given by the following operations:

- **Translation:** a 3D translation vector, ${}^w\mathbf{t}_c$, describing the relative position of the origin of the camera reference frame to the world reference frame.

$${}^w\mathbf{t}_c = [x_T, y_T, z_T]^T \quad (2.6)$$

- **Rotation:** a orthogonal 3×3 rotation matrix, ${}^c_w R$, describing the rotations needed to align the world reference frame with the camera reference frame.

$${}^c_w R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix} \quad (2.7)$$

Assuming a point $\mathbf{P}$ in 3D space, the notation that describes the relation between its coordinates in world frame ${}^w\mathbf{P}$ and camera frame ${}^c\mathbf{P}$ is given by:

$${}^c\mathbf{P} = {}^c_w R ({}^w\mathbf{P} - {}^w\mathbf{t}_c),$$

or,

$${}^c\mathbf{P} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -\mathbf{r}_1 \cdot {}^w\mathbf{t}_c \\ r_{21} & r_{22} & r_{23} & -\mathbf{r}_2 \cdot {}^w\mathbf{t}_c \\ r_{31} & r_{32} & r_{33} & -\mathbf{r}_3 \cdot {}^w\mathbf{t}_c \end{bmatrix} \cdot {}^w\mathbf{P}^* \quad (2.8)$$

$${}^c\mathbf{P} = M_{ext} \cdot {}^w\mathbf{P}^*$$

Where ${}^w\mathbf{P}^* = [{}^wX, {}^wY, {}^wZ, 1]^T$ is a representation of ${}^w\mathbf{P}$ for mathematical convenience and M_{ext} is the extrinsic matrix.

Figure 8 – The perspective camera model with the camera's parameters.

Source: author.

2.2.0.2 Intrinsic Parameters

The set of parameters used for mapping points in the image plane from the camera frame to the image coordinates are called as intrinsic parameters. This item was based on (BROWN, 1965; J.G. Fryer; D.C. Brown, 1986; HEIKKILA; SILVEN, 1997). For the perspective camera model, it is used six sets of parameters, which are also illustrated in Figure 8 and described as follows:

- f : The *focal length*, which is the distance between the FOP and the image plane.
- (o_x, o_y) : The coordinates in pixel of the point $\mathbf{o}'$ where the optical axis intercept the image plane.
- (s_x, s_y) : The effective *size of the pixel* in millimeters.
- (k_1, k_2, k_3) : The *radial distortion parameters* introduced by the optics.
- (p_1, p_2) : The *tangential distortion parameters* caused by the decentering of the lens components.
- α_c : The skew coefficient defines the angle between the x and y pixel axes, because it is not always that the pixel sensor is perfectly rectangular.

Considering a point $[^c x, ^c y]^T$ (in camera reference coordinates) in the image plane of the perspective camera model, one can include radial (d_r) and tangential (d_{tx} and d_{ty}) distortions of the *plumb model* (introduced by Brown (1965)) by the following equation:

$$\begin{aligned}
 ^c x_d &= d_r \cdot ^c x + d_{tx} \\
 ^c y_d &= d_r \cdot ^c y + d_{ty} \\
 d_r &= (1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_3 \cdot r^6) \\
 d_{tx} &= 2 \cdot p_1 \cdot ^c x \cdot ^c y + p_2 \cdot (r^2 + 2 \cdot ^c x^2) \\
 d_{ty} &= p_1 \cdot (r^2 + 2 \cdot ^c y^2) + 2 \cdot p_2 \cdot ^c x \cdot ^c y \\
 r^2 &= ^c x^2 + ^c y^2
 \end{aligned} \tag{2.9}$$

Once distortions are applied, the pixel coordinates $[^i x, ^i y]^T$ can be computed as follows:

$$\begin{aligned}
 ^i x &= -\frac{f}{s_x} (^c x_d + \alpha_c \cdot ^c y_d) + o_x \\
 ^i y &= -\frac{f}{s_y} \cdot ^c y_d + o_y
 \end{aligned} \tag{2.10}$$

or

$$\begin{aligned}
 \begin{bmatrix} ^i x \\ ^i y \\ 1 \end{bmatrix} &= \begin{bmatrix} -f/s_x & -\alpha_c \cdot f/s_x & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} ^c x_d \\ ^c y_d \\ 1 \end{bmatrix} \\
 \begin{bmatrix} ^i x \\ ^i y \\ 1 \end{bmatrix} &= M_{int} \cdot \begin{bmatrix} ^c x_d \\ ^c y_d \\ 1 \end{bmatrix}
 \end{aligned} \tag{2.11}$$

where M_{int} is denominated as the intrinsic matrix.

Now that the extrinsic and intrinsic parameters are explained, it is possible to compute the world-image projection. From Equations (2.5) and (2.8), one can calculate the point projected on the image plane in the camera reference coordinates as follow:

$$\begin{bmatrix} ^c x \\ ^c y \\ f \end{bmatrix} = \frac{f}{^c Z} \cdot M_{ext} \cdot ^w \mathbf{P}^* \tag{2.12}$$

Then, taking $^c x$ and $^c y$ and applying the Equation (2.9), distortions will be added to find $^c x_d$ and $^c y_d$. Finally, the coordinates in pixels of the projected point in the image can be found by multiplying $[^c x_d, ^c y_d, 1]^T$ by the intrinsic matrix, as done in Equation (2.11).

2.3 The Stereo System

On the previous section, it was introduced the perspective camera model, which is commonly applied for monocular cameras. Although the perspective camera model

provides mathematical relation from a 3D point in the environment to its projection in the image, it is difficult to estimate the inverse relation with one camera only. To solve this problem, one can employ a stereo camera, which is going to be explained in this section.

Assuming that there are two monocular vision systems in distinct positions pointing to the same scenario, it is possible to infer the 3D structure from the two captured images. This is feasible for what is called *stereo vision*. Taking the human eyes as an example, both of them will be looking to the same scene, but due to their difference of positions, each of them is going to have different view of the scene. However, because there is an intersection of those views the formed image in the left retina will have correspondent points to the image formed in the right retina. The difference between the position of a point in the left retina and the position of the correspondent point in the right retina defines the disparity of these points, and the disparity of all corresponding points assemble the *disparity map*, which is used by the brain to reconstruct a 3D representation of the viewed scene.

Now, computationally thinking, and replacing the human eyes by monocular cameras, the idea is quite similar. Nevertheless, there are two main problems to be solved, namely *correspondence* and *reconstruction*. The first problem consists in matching the left image with the right image, finding corresponding parts. Although it seems trivial for humans, this task is rather difficult to solve by computational algorithms. The second problem to be solved is the reconstruction of a 3D representation of the scene. After solving the correspondence problem, one can compute the disparities of all the image points to obtain the disparity map. Then, with the geometry of the stereo system, denominated as the epipolar geometry, it is possible to convert the disparity map to a 3D map.

The following sections, which were mostly based on (SZELISKI, 2011; TRUCCO; VERRI, 1998), makes a brief overview about the correspondence problem, and explains the geometric setup of a stereo system.

2.3.1 The Correspondence Problem

If one point on the environment is observed by two cameras, then it will be projected on the images obtained by both sensors. But, considering that the scene can be complex, finding two corresponding points can be very challenging too. Assuming that there is a big intersection between the views of both cameras and that the distances of the observed points from both cameras is much larger than the distance between them, then most points from the scene will be present on both images, and the corresponding regions will be similar. In this case, the correspondence problem consists in finding the left image elements to match, and match them with correspondent elements from the right image according to a similarity measure. The present algorithms to solve this problem can be divided in two big classes, the *sparse* and *dense* correspondence methods (SZELISKI, 2011). For this project, it was chosen the dense correspondence approach, since a dense disparity map is

needed to project the features to a point cloud.

Figure 9 – Visual example of correlation-based matching. The blue correlation window is associated with a pixel $\mathbf{p}_1$ from I_1 , and its equivalent position in I_2 is illustrated in dashed lines. In image I_2 , the algorithm tries to find a matching in the search region represented in yellow. When it is found (in green), the disparity is given by the displacement between the positions $\mathbf{p}_1$ and $\mathbf{p}_2$.

Source: author.

The dense correspondence methods try to match the maximum amount of pixels from one image with the other, including regions without textures or any features type. Currently, the correlation-based method (also denominated as block-matching) are among the most widely used approaches. Basically, they adopt image windows of fixed size as elements to match between the images of both cameras of the stereo system. The correspondence is found by minimizing the cost function of disparities between two elements in a given search region. Figure 9 shows a visual example of how it works.

Table 1 – A summary of the correlation-based correspondence (or block matching) algorithm.

Algorithm 1: Correlation-Based Correspondence

Input : I_1, I_2
Output : disparity map D

```

1 begin
2   for each pixel  $\mathbf{p}_1 = [i, j]^T$  do
3     for each displacement  $\mathbf{d} = [d_1, d_2]^T \in R(\mathbf{p}_1)$  do
4        $cost(\mathbf{d}) = \sum_{m=-L}^L \sum_{n=-L}^L \phi(I_1(i+m, j+n), I_2(i+m-d_1, j+n-d_2))$ 
5     end
6      $D[i, j] = \arg \min_{\mathbf{d} \in R} cost(\mathbf{d})$ 
7   end
8 end
```

Source: Trucco and Verri (1998), Scharstein and Szeliski (2002)

For better comprehension, the algorithm of the correlation-based methods is presented in Table 1. Assume a stereo pair of images I_1 and I_2 , the pixels $\mathbf{p}_1$ and $\mathbf{p}_2$ in these images, a fixed size $2L + 1$ of the correlation window, the search region $R(\mathbf{p}_1)$ in the image

2 associated with $\mathbf{p}_1$, and a cost function $\phi(u, v)$. For each pixel $\mathbf{p}_1$ position in I_1 , it is computed the cost function for correlation windows associated with every pixel in I_2 inside $R(\mathbf{p}_1)$. Then, the pixel $\mathbf{p}_2$ with the minimum correlation cost is chosen as the true match of $\mathbf{p}_1$, and the displacement $\mathbf{d}$ between the position of both pixels is taken as the disparity.

The most popular cost function and the choice for this work is the absolute differences ($\phi(u, v) = |u - v|$), which performs the sum of absolute differences (SAD). Besides them, there are also the sum of squared differences (SSD), cross-correlation, normalized cross-correlation (NCC), and zero mean sum of absolute differences (ZSAD) (HUMENBERGER et al., 2010).

2.3.2 Epipolar Geometry

Considering that a stereo system with two cameras in different poses (usually, positioned side-by-side) pointing to a common scenario, it is possible to find correspondent elements between the pair of stereo images. But, finding those elements may be challenging, since the search region is unknown. However, if the geometry of the stereo system is known, it is possible to simplify the correspondence problem.

Figure 10 – Illustration of the epipolar geometry of an arbitrary stereo system.

Source: author.

So, given a stereo system, the geometry that defines its setup is called *epipolar geometry*. Figure 10 shows the epipolar geometry of an arbitrary stereo system. Assuming two cameras in different positions and pointing to the same scene, the plane formed by a point P from the environment and the FOPs F_l and F_r from the left and right cameras respectively is called *epipolar plane*, and the lines where this plane intersects the image planes are the *epipolar lines*. Considering the illustrated configuration, it is possible to observe that one of the extreme point from a epipolar line is actually the projection of the FOP from the other camera, and this point is denominated as *epipole*.

The epipolar geometry can be defined by intrinsic and extrinsic parameters of the stereo system, which define the position and orientation of a camera in relation to

the other. In practice, these parameters can be estimated through a calibration process. For instance, the calibration method proposed by Zhang (2000) is widely employed, and it is already implemented in some popular open-sourced computer vision libraries (e.g. OpenCV).

2.3.2.1 Rectification

Previously, some matching methods were presented, but in stereo systems there are additional information about the cameras positions that can be used to improve the solution of correspondence problems. So, if the epipolar lines from the left and right images were collinear and parallel to the horizontal axes of the images, as shown in Figure 11, the matching task would be reduced to a scanline in just one dimension. These properties can be obtained by the *rectification*.

Figure 11 – Rectification of a stereo pair.

Source: author.

Assuming that the extrinsic parameters are known, one can have the relation between the left and right cameras, that is given by a translation vector ${}^l\mathbf{t}_r$ and a rotation matrix rR_l . And with those, it is possible to compute the rectification. Firstly, the left camera is rotated so that the resulting epipole goes to infinity along the horizontal axis of the image frame. This rotation is defined by:

$$R_{left} = R_{rect} = \begin{bmatrix} \mathbf{e}_1^T \\ \mathbf{e}_2^T \\ \mathbf{e}_3^T \end{bmatrix}, \quad (2.13)$$

where

$$\begin{cases} \mathbf{e}_1 = \frac{{}^l\mathbf{t}_r}{\|{}^l\mathbf{t}_r\|} \\ \mathbf{e}_2 = \frac{1}{\sqrt{x_T^2 + y_T^2}} \begin{bmatrix} -y_T \\ x_T \\ 0 \end{bmatrix} \\ \mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2 \end{cases} \quad (2.14)$$

Then, the same rotation is applied to the right camera frame followed by the rotation ${}^r_l R$. Mathematically,

$$R_{right} = {}^r_l R R_{rect}. \quad (2.15)$$

Finally, an adjustment on the scale of both camera frames is done by the following formula:

$$\mathbf{p}'_{side} = \frac{f}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}, \text{ with } \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_{side} \begin{bmatrix} x_{side} \\ y_{side} \\ f \end{bmatrix} \quad (2.16)$$

where *side* is replaced by *left* or *right* in each image.

2.3.2.2 Reconstruction

After doing rectification and finding correspondent elements, the next step would be the 3D reconstruction. However, there are three distinct situations for this problem, which depends on how much information about stereo parameters is known. For instance, it will be considered that both intrinsic and extrinsic parameters are known. In this case, the reconstruction problem can be solved by triangulation.

Although in theory the 3D point reconstructed from the left and from the right image should coincide, in practice the noise prevents it to occur. So, the ray l that starts from F_l and pass through the image point ${}^l\mathbf{p}$ does not intersect with the ray r . Note that during the reconstruction, there is no epipolar plane, because there is no point $\mathbf{P}$ where r and l intersects. So, the reconstructed 3D point will be localized at the medium point between the rays, as shown by Figure 12.

Let $a^l\mathbf{p}$ be the vector that starts from F_l and pass through the left image point, and similarly for $b^r\mathbf{p}$ in the case of right side of the stereo system. Taking the left frame as reference, the vector from the right image point is expressed as ${}^l\mathbf{t}_r + b({}^r_l R^T \cdot {}^r\mathbf{p})$. Moreover, let $\mathbf{w} = c[{}^r\mathbf{p} \times ({}^r_l R^T \cdot {}^r\mathbf{p})]$ be the vector representing the distance between the rays l and r . From here, the problem is reduced to determining the midpoint $\mathbf{P}'$ of $\mathbf{w}$.

The solution can be achieved by solving the linear system of equations given by:

$$a^l\mathbf{p} - b({}^r_l R^T \cdot {}^r\mathbf{p}) + \mathbf{w} = {}^l\mathbf{t}_r. \quad (2.17)$$

Finally, the reconstructed 3D point is computed by:

$${}^l\mathbf{P}' = a^l\mathbf{p} + \frac{\mathbf{w}}{2} \quad (2.18)$$

Figure 12 – 3D reconstruction by triangulation with nonintersecting rays.

Source: author.

2.4 Occupancy Grid Maps

Until here, all the background concepts applied for road marking detection in 3D space proposed by this work were already introduced. From now, basic concepts necessary for localization are going to be explained, and in this section, mapping concept and range sensor model are covered.

Currently, there are two types of metric representation of maps: the *Occupancy Grid Maps* (OGM) (MORAVEC, 1988; ELFES, 1989), and the maps based on *Landmarks* (FUENTES-PACHECO; RUIZ-ASCENCIO; RENDÓN-MANCHA, 2012, p.69). But OGM is the most used due to its precision on indicating the occupied and free spaces (THRUN, 2002, p.1). The explanations in this section were based on (THRUN; BURGARD; FOX, 2006; HATA, 2016).

Initially, to each grid cell m_i on OGM a binary occupancy value was assigned to specify if the correspondent location was occupied with an object. However, generating this type of map requires exteroceptive sensors (e.g. SONAR and LIDAR), which are subjected to noise and uncertainty during measurements. For this reason, probabilistic techniques

had been applied. So, instead of binary values, each position cell is assigned by a value corresponding to the occupancy probability of its associated area on the environment.

Formally, given the position $x_{1:t}$, the sensor measurement $z_{1:t}$, and time stamp t , the occupancy probability of the map is defined as:

$$p(m|x_{1:t}, z_{1:t}) = \prod_{x,y} p(m_i|x_{1:t}, z_{1:t}) \quad (2.19)$$

The odds ratio of the occupancy probability of a cell can be obtained by Bayes filter, as follows:

$$\frac{p(m_i|x_{1:t}, z_{1:t})}{1 - p(m_i|x_{1:t}, z_{1:t})} = \frac{p(m_i|x_t, z_t)}{1 - p(m_i|x_t, z_t)} \cdot \frac{p(m_i|x_{1:t-1}, z_{1:t-1})}{1 - p(m_i|x_{1:t-1}, z_{1:t-1})} \cdot \frac{1 - p(m_i)}{p(m_i)} \quad (2.20)$$

From this equation, it can be seen that the measurement updates are computed by subsequent multiplications of probabilities. But, in cases when the probabilities are close to 0 and 1, it could have truncation problems. To avoid it, one can apply the log odds ratio, given by:

$$l(x) = \log \frac{p(x)}{1 - p(x)} \quad (2.21)$$

So, applying logarithm to the Equation (2.20):

$$\begin{aligned} \log \frac{p(m_i|x_{1:t}, z_{1:t})}{1 - p(m_i|x_{1:t}, z_{1:t})} &= \log \frac{p(m_i|x_t, z_t)}{1 - p(m_i|x_t, z_t)} + \log \frac{p(m_i|x_{1:t-1}, z_{1:t-1})}{1 - p(m_i|x_{1:t-1}, z_{1:t-1})} + \log \frac{1 - p(m_i)}{p(m_i)} \\ &= \log \frac{p(m_i|x_t, z_t)}{1 - p(m_i|x_t, z_t)} + \log \frac{p(m_i|x_{1:t-1}, z_{1:t-1})}{1 - p(m_i|x_{1:t-1}, z_{1:t-1})} - \log \frac{p(m_i)}{1 - p(m_i)}, \end{aligned}$$

it results on the log odd ratio of the cell occupancy:

$$l_{i,t} = \log \frac{p(m_i|x_t, z_t)}{1 - p(m_i|x_t, z_t)} + l_{i,t-1} - \log \frac{p(m_i)}{1 - p(m_i)} \quad (2.22)$$

In this way, the measurement update becomes computationally more practical, involving only an addition to the prior probability. Moreover, to recover the occupancy probability of the cell m_i , one can simply apply:

$$p(m_i|x_{1:t}, z_{1:t}) = 1 - \frac{1}{\exp(l_{i,t})} \quad (2.23)$$

2.4.1 Likelihood Fields

For mapping and localization tasks, range sensors are frequently used (e.g. Sonar and Lidar). But, as sensor measurements are caused by physical objects instead of the map of those objects, sensor models are necessary to condition sensors to the map. There are two common sensor models, namely *Beam Model* and *Likelihood Fields*. The first one approaches the physical model of range sensors. Nevertheless, this model suffers lack of

smoothness due to small obstacles, and it is computationally complex because of the ray casting. Furthermore, it cannot be applied for stereo cameras. On the other hand, the second model does not have a plausible physical explanation, so it can be addressed to any sensor that is able to make metric measurements. Moreover, it overcomes the Beam model limitations, providing smoother maps and measurements. For this reason, the Likelihood Fields is preferred in practice (THRUN; BURGARD; FOX, 2006).

Figure 13 – Illustration of the perception model. For simplicity, the illustration considers that the z axis from every reference frames are parallels.

Source: author.

The key idea of Likelihood Field is to project the sensor measurement point to the map coordinates. From here, the explanations can be followed taking the Figure 13 as illustration.

In the case of using stereo camera as sensor, one can use the Equation (2.18) to get the measurement point, ${}^c\mathbf{z}$, in relation to the left camera frame. If the vehicle frame, given by $({}^v x, {}^v y, {}^v z)$, is taken as the reference for extrinsic parameters of the left camera, then it is possible to compute the relative position of the measured point on the coordinate system of the vehicle, ${}^v\mathbf{z}$, by:

$${}^v\mathbf{z} = {}^vR \cdot {}^c\mathbf{z} + {}^v\mathbf{t}_c \quad (2.24)$$

Then, for a given position of a vehicle on the map coordinate system $({}^m x, {}^m y, {}^m z)$, ${}^m\mathbf{t}_v$, and the orientation, given by mR , one can define the measured point on the map coordinate system by:

$${}^m\mathbf{z} = {}^mR \cdot {}^v\mathbf{z} + {}^m\mathbf{t}_v$$

or,

$${}^m\mathbf{z} = {}^mR \cdot ({}^vR \cdot {}^c\mathbf{z} + {}^v\mathbf{t}_c) + {}^m\mathbf{t}_v \quad (2.25)$$

Every sensor measurements are subjected to imprecisions caused by noises, failures, or random measurements. One way to deal with it is by employing the Gaussian probability density function on measurements as follows:

$$p_{hit}(d^2, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left(-\frac{d^2}{2\sigma^2}\right), \quad (2.26)$$

where d is the distance to the closest obstacle, and σ is the uncertainty assigned to the mapped obstacle.

During localization, the Likelihood Fields model can be used to match the measurement obtained by a sensor with the map by estimating the vehicle's pose that maximizes the product of all measurement probabilities given by Equation 2.26. However, searching the nearest obstacle in the map for every measurement is computationally very costly. But, in practice, one can pre-compute the Likelihood Field in the map. By this way, during localization, the likelihood calculation would be simplified to a simple and fast table lookup.

Figure 14 – (a) Example of a random environment, and (b) its Likelihood Field representation. The brighter a location, the higher is the likelihood value.

(a) Example environment.

(b) Likelihood field.

Source: Thrun, Burgard and Fox (2006).

2.5 Monte Carlo Localization

Now that OGM (map model) and the Likelihood Field model (range sensor model) were explained, in this last section of the chapter covers the localization model employed in this work.

Since the Monte Carlo localization (MCL) is easy to implement and is robust with uncertainties, it is widely applied for Markov localization (THRUN et al., 2001). The MCL algorithm estimate the robot pose (position and orientation) according to its movements and external sensing from exteroceptive sensor data, representing the posterior belief as particles. This is because MCL is a localization algorithm based on particle filters.

As reported by Thrun, Burgard and Fox (2006), MCL consists of three basic steps, namely prediction, updating, and resampling. Given a motion model, the first step predicts the next particles poses. Then, according to the measurement model it updates the belief and their respective importance weights. Finally, the resampling step creates a new set of particles with respect to the importance weights from the updating step.

The idea of this algorithm is to represent the belief of the robot pose, $bel(x_t)$, by a set of M particles $\mathcal{X}_t = \{x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]}\}$. Therefore, the probability of the robot be in a certain position will be proportional to the density of particles located there. Initially, the pose is unknown, so the particles are distributed equally throughout the map. But, after some iterations the ambiguities are reduced and the belief starts to converge.

Table 2 – Monte Carlo Localization, an algorithm based on the particle filters.

Algorithm 2: Monte Carlo Localization	
<hr/>	
Input	$\mathcal{X}_{t-1}, u_t, z_t$
Output	$\mathcal{X}_t$
1	begin
2	$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset;$
3	for $m = 1$ <i>to</i> M do
4	sample $x_t^{[m]} \sim p(x_t u_t, x_{t-1}^{[m]});$
5	$w_t^{[m]} = p(z_t x_t^{[m]});$
6	$\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle ;$
7	end
8	for $m = 1$ <i>to</i> M do
9	draw i with probability $\propto w_i^{[m]};$
10	add $x_t^{[i]}$ to $\mathcal{X}_t;$
11	end
12	end

Source: Thrun, Burgard and Fox (2006)

For better comprehension, the algorithm in Table 2 will be described step-by-step. Suppose that in a moment t the robot performs a motion u_t and get sensor data z_t from the environment, and that the particles set resulted from the last iteration is given by $\mathcal{X}_{t-1}$. Having these information, the iteration starts by predicting the next pose for each particle according to the **motion model**, given by $x_t^{[m]} = p(x_t|u_t, x_{t-1}^{[m]})$ (line 4). After predicting new poses, it updates the belief by assigning importance weights to

every particle (line 5). The importance weight $w_t^{[m]}$ is nothing more than the probability of having the measurement z_t given the position $x_t^{[m]}$ of the m -th particle given by a **measurement model**, e.g. Likelihood Fields for range finders. Then, the predicted poses and their respective weights make the *priori* belief, represented by the $\bar{\mathcal{X}}_t$ set.

At last, the particles set is resampled (line 8 to 11) with different distribution, but keeping the same size M . This operation is done by drawing particles from $\bar{\mathcal{X}}_t$ with probability proportional to the importance weights. Since sampling is done with replacement, the resulting set is going to have replicated particles and some others with low importance are going to be removed. As consequence, there are going to be higher concentrations in regions where previous particles had bigger importance weights.

Figure 15 – Example of two consecutive iterations of MCL.

Source: adapted from (THRUN; BURGARD; FOX, 2006).

Figure 15 shows a simple example of MCL running two complete iterations in 1D space. It starts with a uniform distribution of particles and evenly weighted (15(a)). At first, there are no motion, so it jumps to the importance weights update. In this step (15(b)), the robot senses a door, so the measurement model produces the probability distribution $p(z|x)$, which shows peaks of probability in places with doors. Then, it updates the weights of the particles in $bel(x)$. As the final step of an iteration, it resamples the particles according to the weights and reinitialize them with uniform importance (15(c)). On the second iteration, the robot moves and the first step predicts the particles poses (15(d)). Then, the robot senses a door again and the measurement model generates the same distribution $p(z|x)$, updating the weights (15(e)). Finally, the resampling results in a new distribution with higher concentration of particles on the second door, where the

robot is located (15(f)).

The non-parametric nature of MCL allows it to approximate to almost any distribution of importance, including complex and multimodal distributions. However, it is important to emphasize about two issues. First, the number of particles to be used influences the algorithm performance, i.e. the algorithm gets more accurate with larger number of particles, but also becomes slower due to higher computational cost. The other issue is about wrong convergence. After particles get concentrated in a small cluster, it can't split. In an environment with a lot of similar features, the algorithm may converge to a wrong pose. However, aiming to solve both problems, Fox (2002) proposed an augmented version of MCL with the Kullback-Leibler divergence sampling (KLD-sampling) to avoid erroneous convergences and to adapt the number of particles according to the particles distribution. In this way, it reduces the number of particles when the distribution is already concentrated, and scatters the distribution with additional random particles when it is too concentrated.

3 RELATED WORKS

The following sections review three topics related to this project, namely road features detection, visual approaches for metric localization, and metric localization. The first topic explores the algorithms used for detecting curbs and road markings as road features, and also the techniques to segment the road scenarios that could be used for this purpose. Then, in the second section, a survey of existing techniques for localization based on cameras as sensors is presented, including an analysis of the state-of-the-art. Finally, on the metric localization section, the main topics that defines a metric localization are presented and reviewed.

3.1 Road Features Detection

3.1.1 A Survey

Figure 16 – The four main modules for features detection.

Source: author.

In literature, most road marking detection approaches focus on lane markings only, and the methods for their detection are similar to curb detection. In both cases (lane markings and curbs), the detection systems are mainly formed by four modules: image pre-processing, features extraction, model fitting, and time integration. Figure 16 shows a block diagram with those modules. The first two modules usually consist of image processing and computer vision techniques to detect the regions of the image that are most likely to contain features (in this case, lane markings or curbs), while the third is the module that fits the features to the lane or road model. Finally, the last module, the time integration, takes the previous frame and the motion information into account to validate the features hypotheses.

3.1.1.1 Image Pre-processing

On the first module, the image pre-processing is applied to remove (or reduce) undesired imaging effects (like image misleading by illumination and weather conditions) and irrelevant image parts. There are two ways of handling the imaging effects, by image enhancement and by gradually adjusting the camera iris or gain, according to the lightning condition of the environment. However, the latter technique still cannot deal with abrupt

illumination change (e.g. entering or exiting a tunnel). On the other hand, image parts pruning have been employed to manage irrelevant image regions.

Currently, the major source of clutter on images are strong shadows, which cause confusion on features detection systems due to their intensity edges. Some authors have been trying to overcome the illumination-related effects on images by exploring transformation over color spaces. For example, Álvarez, López and Baldrich (2007) proposed a transformation of the RGB color space to create images with low effects of shadows. Also, in (KATRAMADOS; CRUMPLER; BRECKON, 2009), a combination of three different color spaces (HSL, YCbCr and LAB) was done to derive four illumination invariant image spaces. Alternatively, Du and Tan (2016a) used the Hue channel of HSI representation as the input data of the system. The most important fact about the last two approaches presented here is that they were able to robustly deal with shadows (as shown in Figure 17), although they still do not fulfill the necessities of this project (road markings and curbs detection).

Figure 17 – Examples of approaches used to reduce the illumination effects on images. The first column shows the original images; the second shows the chroma map proposed by Katramados, Crumpler and Breckon (2009); and the third shows the hue channel of HSI representation.

Source: Image samples taken from the CaRINA dataset (SHINZATO et al., 2016), and image processing done by the author.

Other important factors that cause ambiguities during features detection are the irrelevant image parts, which can be caused by obstacles or non-road regions (e.g. scene above the horizon). To avoid detecting obstacles as features, they have been removed from the image. In (JO et al., 2015), three laser scanners were used to detect surrounding objects and remove them from the camera images before detecting the features. Moreover,

to prevent excessive computations and false features detection, some researchers have been defining regions of interest (ROI), selecting only the region of the image where is most likely to contain features. For example, Zhang et al. (2009) removed the upper half of the image and used only the bottom half as the ROI. However, this proportion may change in accordance with the camera pose on the vehicle.

3.1.1.2 Features Extraction

Once the enhancement and ROI determination of the image are done, the features can be more reliably extracted. On the module of features extraction, there are actually two different types of features to be detected, namely road markings and curbs. For each type, different approaches are used.

Since road markings are actually made to guide human driving by visual perception, they are always drawn in bright colors in order to form a strong contrast with the dark road surface. Taking this observation into account, most authors proposed approaches for road markings detection by pattern of a low-high-low brightness values (SEBSADJI et al., 2010; LIU et al., 2011; HILLEL et al., 2014). In the literature, there are commonly used techniques to extract edges from local brightness variance by local gradient followed by a thresholding. But, because clutter is present on images, even with pre-processing, large number of outliers are obtained.

Figure 18 – Examples of lane markings detection performances in case of using (a) top-hat filter and (b) gray-level ridges.

(a) Top-hat filter.

(b) Gray-level ridges.

Source: Veit et al. (2008), Du and Tan (2016b).

For this reason, more advanced approaches based on gradient have been proposed. In (VEIT et al., 2008; JO et al., 2015), a top-hat kernel is used to detect road markings by a matched filter on the bird's-eye view image. Alternatively, Du and Tan (2016b) proposed a robust method that extract lane markings by detecting low-high-low patterns in shape

of gray-level ridges. The key idea of the gray-level ridge concept is that the intensity of a pixel in gray-scale increases as it gets closer to the center axis of a lane line. Although these techniques have good performance on lane markings detection in cases with good illumination, they still suffer with shadows. Also, they were designed to handle with lane markings only, which usually are present on regions without intersections of streets. Figure 18 shows the performance of these approaches.

To detect other road markings (e.g. zebra crossing, speed bump, horizontal warning letter, etc) different techniques would be necessary. For example, Du and Tan (2016a) used the gray-level ridge filter for lane markings, a method based on Fast Fourier Transform (FFT) for zebra crossing and speed bump detection, and Principal Component Analysis (PCA) for warning letters detection.

In relation to curbs detection, some authors have been using the three-dimensional structure to detect the curbs according to their step-shape. However, most of them used the 3D information taken from LIDAR data, since it is more precise (HUANG et al., 2009; HERNANDEZ; MARCOTEGUI, 2009; JO et al., 2015; HATA; WOLF, 2016). But there are some works based on stereo vision that tried to detect curbs by their shape. In (ONIGA et al., 2007), an elevation map computed from dense stereo was used to detect the elevation caused by curbs. In (PRADEEP; MEDIONI; WEILAND, 2008), a clustering of planes from a 3D point cloud was done to find the curb as the transition between two planes, as shown in Figure 19. But, in both cases, the approaches were applied for nearby curbs. Because disparities get smaller as the distance increases, the depth resolution decreases and it becomes harder to detect curbs with small elevations.

Figure 19 – Curb detection according to its step-shape.

Source: Pradeep, Medioni and Weiland (2008).

Other option that could be employed is detecting curbs by firstly estimating the road area. There are many papers that have been published for road estimation (ÁLVAREZ; LÓPEZ; BALDRICH, 2007; KATRAMADOS; CRUMPLER; BRECKON, 2009; SHINZATO et al., 2012; SHINZATO; GOMES; WOLF, 2014; MENDES; FREMONT; WOLF, 2016), and the state-of-the-art approaches can be checked on the KITTI Vision Benchmark Suite¹, which was introduced by Fritsch, Kuhn and Geiger (2013). This kind of approach is actually more useful for safe navigation and obstacle avoidance.

¹ Source: GEIGER, A. et al. **Road/Lane Detection Evaluation 2013**. Available from Internet: <http://www.cvlibs.net/datasets/kitti/eval_road.php>. Accessed in: 1 apr. 2017.

Notwithstanding, considering that the boundaries of a road in urban areas are usually limited by curbs, road estimation could be used as a tool to detect curbs. But, in cluttered scenarios with many occlusions by parked vehicles, the curb detection on road area would be challenging.

3.1.1.3 Model Fitting

Since urban scenarios are usually cluttered, even if using the state-of-the-art features extractor, there will be noisy detections and outliers (false detections). One way to overcome this problem is by fitting the features detected from the image to a geometric model, and use the fitted model as feature. Alternatively, the model fitting can be used only for filtering the outliers, and use only the inlier points as features. Recently, many parametric models have been used to support the features extraction process. For example, hyperbola curves have been used by (DU; TAN, 2016a; DU; TAN, 2016b), parabola by (HUANG et al., 2009; LU et al., 2014), and a linear-parabolic model by (LIU et al., 2011).

To fit the parametric model to the detected features, the most popular approach is the Random Sample Consensus (RANSAC) (HUANG et al., 2009; DU; TAN, 2016b). However, there are other techniques that can be used, like least squares optimization (LIU et al., 2011) and least trimmed squares (LTS) (HATA; WOLF, 2016). Although the model fitting is an important tool to improve features extraction, some authors do not employ it, and instead, they use only time integration to estimate features positions before detecting it (ENZWEILER et al., 2013; WOLCOTT; EUSTICE, 2014).

3.1.1.4 Temporal Integration

In localization problems (further discussion in Section 3.3.4), the knowledge of previous position of the vehicle is important to filter wrong estimations and to have good predictions. Similarly, the same idea of time integration can be applied to features detection in order to improve performance. Recently, this approach have been used (ENZWEILER et al., 2013; SCHREIBER; KNÖPPEL; FRANKE, 2013; LU et al., 2014; WOLCOTT; EUSTICE, 2014).

The main benefits that the temporal integration can provide to features extraction are: accuracy improvement, less computational cost, and noise filtering (HILLEL et al., 2014). The accuracy of the features extraction can be improved by predicting the features position before detecting it, which also reduces the computation. Moreover, because there is an initial guess of the features positions, outliers can be easily removed. However, the method used for features detection must be robust since the first iteration.

3.1.2 Deep Learning for Road Scenes Segmentation

Even though there are many works already developed for road marking and curb detection, deep learning has increasingly been applied for computer vision problems, showing to be an attractive approach.

Deep learning is a term addressed to the problem of assigning values to parameters of neural network architectures with many hidden layers, and the heart of recent advances in this field are the CNN architectures (SCHMIDHUBER, 2014; DUMOULIN; VISIN, 2016) along with more powerful GPUs. This type of architecture can be applied for any application related to pattern recognition on images, and have been achieving the state-of-the-art performances on character recognition (LECUN et al., 1989), ImageNet² classification (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), pedestrian detection (CAI et al., 2016), road estimation (TEICHMANN et al., 2016; MENDES; FREMONT; WOLF, 2016; CHEN; CHEN, 2017) and scene segmentation (BADRINARAYANAN; KENDALL; CIPOLLA, 2017).

Initially, every CNN architectures followed the same conventional pattern of architecture, which consists of an alternating sequence of convolutional and pooling layers, finalized with few MLP layers (also denominated as totally connected layers). But, this architecture setup does not allow a pixelwise classification (or image segmentation), i.e. it gives only one classification for the entire input image. Therefore, early CNN-based approaches for image segmentation employed conventional architectures to classify sliding windows (LI; ZHAO; WANG, 2014). Nonetheless, there is a disadvantage on employing this technique, which is the inference computation for each window position on the input image.

Aiming to overcome this problem, some authors have proposed solutions that can segment the image in a single inference. For example, Mendes, Fremont and Wolf (2016) trained a conventional CNN architecture to classify patches of images, then converted the last fully connected layers into convolutional layers during test. In this way, each pixel of the output feature maps corresponds to the classification of an entire patch of the input image. By doing this, the architecture became a FCNN, and every window positions could be classified at once. However, if precision is required, this method may not be sufficient due to its limitation to the patch size.

Alternatively, more recent works have proposed the use of deconvolution (or transpose convolution) (ZEILER et al., 2010) to upsample features computed by previous sequence of convolutional and pooling layers. In this way, it is possible to create FCNN in an encoder-decoder architecture (DUMOULIN; VISIN, 2016). A deep architecture of this type can encode location and contextual information to resolve the pixelwise segmentation

² Observation: ImageNet is a dataset for benchmark on image classification introduced in (DENG et al., 2009), and it is available in: <<http://image-net.org/>>.

Figure 20 – MultiNet performance on road estimation. The green plot is the resulting estimation.

Source: Teichmann et al. (2016).

Figure 21 – SegNet performance on urban scenes segmentation for day and dusk test samples.

Source: Badrinarayanan, Kendall and Cipolla (2017).

(LONG; SHELHAMER; DARRELL, 2015).

For this project, the encoder-decoder CNN architecture is specially interesting, since the objective is to detect curbs and road markings in pixel level. Also, considering that these features to be detected are road-related features, it is interesting to note the performances obtained by the state-of-the-art of road estimation and scene segmentation previously cited. For road estimation, Teichmann et al. (2016) proposed a unified encoder-decoder architecture, named MultiNet. In Figure 20, it is possible to observe the robustness of this method, even in images with some shadows. Also, for scene segmentation, Badrinarayanan, Kendall and Cipolla (2017) proposed another FCNN architecture, the SegNet. The robustness against illumination conditions and cluttered urban scenarios of this approach can be noticed in Figure 21. Furthermore, in both cases, intersections of streets are handled reliably.

Although those approaches (MultiNet and SegNet) were not trained for detecting

road markings and curbs, they can be re-trained for this purpose. Since CNN operates with convolutional kernels over the input image and the consecutive feature images, it actually works as many filters in sequence. Still, as the weights of kernels are the CNN parameters, the kernels can be adapted during the training time to extract only the most relevant features for detecting road markings and curbs. Moreover, encoder-decoder architectures allows jointing multiple functionalities. For example, in MultiNet architecture cited before, Teichmann et al. (2016) used the output from the encoder as the input of the decoder network for road estimation, and also as the input of networks for image classification and object detection. Considering that an autonomous navigation system consists of several perception subsystems, having a way to integrate all of them may be a key point to the development of autonomous vehicles.

3.2 Visual Approaches for Metric Localization

When using cameras as sensors for localization, the structures used as map features must be detected on images, and with a stereo configuration or motion information, metric data can be extracted. Given a scenario with static structures, there are two types of information that can be detected from the image, namely landmarks and salient features. Salient features (or just features) are points or regions from the image described by their distinctive appearance. On the other hand, landmarks are regions from the real world described by its 3D appearance. Since landmarks can be projected to the image plane, they also can be detected as salient features (FUENTES-PACHECO; RUIZ-ASCENCIO; RENDÓN-MANCHA, 2012). But, for convenience, salient features correspondent to landmarks will be referred as landmarks in this document.

When using monocular camera for metric localization, motion information is commonly used to get 3D structure of the scene. To solve this problem, monocular visual odometry methods (NISTER; NARODITSKY; BERGEN, 2004; NEWCOMBE; LOVE-GROVE; DAVISON, 2011; FORSTER; PIZZOLI; SCARAMUZZA, 2014; CASELITZ et al., 2016) try to simultaneously determine the camera pose for each frame and the 3D position of a salient feature. For instance, sparse features have been adopted, such as Harris corners (HARRIS; STEPHENS, 1988), Scale Invariant Feature Transform (SIFT) (LOWE, 1999), Features from Accelerated Segment Test (FAST) (ROSTEN; DRUMMOND, 2006), and Speeded Up Robust Features (SURF) (BAY; TUYTELAARS; Van Gool, 2006), not to mention the techniques based on direct appearance correlation, like image patches (or blocks) (FUENTES-PACHECO; RUIZ-ASCENCIO; RENDÓN-MANCHA, 2012). However, this type of solution is usually applied for tracking problems and SLAM, not including the global localization (AQEL et al., 2016).

Besides using motion information for monocular camera approaches, some authors try to find alternative solutions by using specific structures on road scenes as landmarks,

such as all the road markings (WOLCOTT; EUSTICE, 2014) and lane markings only (CUI; XUE; ZHENG, 2016; XIAO et al., 2016). Wolcott and Eustice (2014) proposed a visual localization system based on a 3D map generated by a LIDAR sensor (Velodyne HDL-32E). In this method, it is assumed the knowledge of the camera parameters (based on the projective camera model) and the estimated initial position (by GPS). By these assumptions, different camera views are predicted from the 3D map according to the projective camera model. So, the localization problem is simplified to a direct match between the obtained image and the predicted view with higher correspondence (by Shannon entropy minimization), which is associated with the most likely position of the vehicle. As result, the achieved performance stayed within $\pm 30\text{ cm}$ of ground truth, running at a rate of 10 Hz .

More recently Xiao et al. (2016) proposed a method where the vehicle position is estimated by matching lane markings detected from the image with a prior 2D map built with LIDAR (Velodyne HDL-64E). In this method the lane markings are detected from the image by simple segmentation technique and projected to the vehicle coordinate system for matching. For localization, the particle filtering is applied, using a GPS for initialization and an IMU data for local motion model. As ground truth, they used a LIDAR based localization result. This approach achieved a performance with $49,5\text{ cm}$ of mean error, varying between 40 cm to 70 cm . However, they have stated rather big longitudinal error on straight roads and good lateral precision, saying it to be qualified for lane-level localization.

In the same year, Cui, Xue and Zheng (2016) proposed a real-time localization approach based on GPS refined with monocular camera fused with IMU data that uses lane markings as reference, using Google maps to obtain the local map. This method consists in estimating the global position by GPS followed by a refinement step, which is addressed to the lane markings detection and matching with the map. To match the detected lane markings with the map, they assumed the road to be a flat surface, and used a simple projective relationship between the image plane and the ground surface to get an inverse perspective mapping. Because of this, the matching operation works fine in flat regions only, achieving a mean absolute error of 15 cm for lane marking detection. According to the authors, the localization system reaches the same lateral accuracy as the lane marking detection, operating at 20 Hz .

Rather than using monocular camera, other authors prefer stereo cameras due to their property of providing the depth information (SCHREIBER; KNÖPPEL; FRANKE, 2013; DU; TAN, 2016a). Although Schreiber, Knöppel and Franke (2013) do not specifies the operation frequency, they reported a localization method that operates in real time and achieves an accuracy of approximately 10 cm . In this approach, a precise map is previously obtained by LIDAR (Velodyne HDL-64E) fused with high resolution camera

setup and manual adjustments, where road markings, boundaries, and curbs are used as the map features. During the localization, a stereo camera is used to detect those features by oriented matched filter (dark-light-dark), project them into the map plane, and match with the map. The tracking problem was the focus of this method, then a GNSS was used to estimate the initial position, and an IMU for motion information.

Alternatively, a more recent technique (DU; TAN, 2016a) proposed a real time lane-level localization method that accurately estimates the vehicle position relative to the lane markings. Although the problem solved by this method is neither a global localization nor a map-aided localization, it could be used as a tool to solve them too. In this approach, the lane markings detector proposed is robust against illumination variations. As result, the maximum estimation error of the vehicle position in comparison with lane markings was 16 *cm* and 26 *cm* in daytime and at night respectively. Comparing with the method proposed by Cui, Xue and Zheng (2016), it had better performance.

Some other more complex visual based localization systems have been proposed, combining more than two cameras in just one localization system. For example, Ziegler et al. (2014) proposed fusing the results from a monocular based localization system with a stereo camera based system. Although its high performance, it depends on three cameras (one camera on the rear part of the vehicle and a stereo configuration on the front part) and two separate computer systems. Also, even more complex system was proposed by Jo et al. (2015), which used a configuration with 6 cameras and 4 laser sensors. But, one thing in common between these two approaches is that both of them adopt road markings as landmarks.

From here, it is notable that the use of road markings in general (lane markings, zebra-crossing, horizontal signalizations, etc) and curbs as road features are preferred for localization systems based on any camera configuration (monocular, stereo or multi-

Table 3 – Existing visual approaches for metric localization.

Camera setup	Reference	Map	Features	Filtering	GPS +IMU	Absolute error (cm)			Rate (Hz)
						Lateral	Long.	Total	
Mono	Wolcott et. al. (2014)	LIDAR map (3D)	Road markings	Kalman Filter	Yes	-	-	30	10
	Cui et. al. (2016)	Google Maps (2D)	Lane mark.	-	Yes	15	-	-	20
	Xiao et. al. (2016)	LIDAR map (2D)	Lane mark.	Particle Filter	Yes	49.5	-	-	Real time
Stereo	Schreiber et. al. (2013)	LIDAR map (2D)	Road mark., boundaries and curbs	Kalman Filter	Yes	-	-	10	Real time
	Du et. al. (2016)	None	Road mark.	Particle Filter	No	2.9±3.9	-	-	9.09
Multi	Ziegler et. al. (2014)	Topolog. map LIDAR map (2D)	DIRD descriptor Road mark., boundaries and curbs	Kalman Filter	Yes	-	-	10	10
	Jo et. al. (2015)	OSM (2D)	Road mark.	Particle Filter	Yes	7.64	52.72	53.65	33.33

cameras setup), as shown in Table 3. The reason for this is that these road features are the most salient features present on roads, and hardly suffer changes along the time (XIAO et al., 2016; HATA; WOLF, 2016). Moreover, since the precision of projection from image degrades with longer distance structures, it is better to take static structures that can be seen as near as possible from the vehicle.

Another observation about vision based metric localization is about the difference between monocular and stereo approaches. On monocular approaches getting metric information is a challenging task, and there are two ways of doing this, by motion information or by assumptions about the road model. To get metric information from motion, one can adopt visual odometry techniques or IMU sensor to estimate motions, and compute the depth values by taking consequent frames as different views of the same scene. But, this kind of approach suffers from scale uncertainty provided by the lack of exact extrinsic parameters of the views (TRUCCO; VERRI, 1998; AQEL et al., 2016). The other way of getting metric information is by projecting the image points to the road surface. For this, one can assume the road surface being a flat plane. Since roads are slightly rounded by sideways and peaks and valleys are present on urban scenarios, errors are predictable for this approach. Besides, on stereo vision approaches metric information can be reliably computed from the disparity map. Consequently, higher accuracy can be achieved compared to approaches that use monocular cameras.

Although metric information is usually needed to solve metric localization problems, Wolcott and Eustice (2014) proposed an interesting method based on monocular camera that does not use metric information explicitly. Instead of projecting the obtained image points to a assumed flat surface for map matching, they adopted the inverse way, projecting 3D points from the map to the image plane for image matching. By this way, they could estimate the position of the vehicle with high precision, reaching a state-of-the-art performance on monocular camera approaches. However, a precise 3D maps is needed for this method, which would require a larger memory capacity in case of wider urban areas if compared with 2D maps used on the state-of-the-art techniques.

Lastly, considering the overall performance, approaches based on stereo vision and multi-cameras setup are better options than methods based on monocular camera. However, considering the complexity of the system, there is an advantage of stereo vision over multi-cameras approaches. Thus, for this project, a single stereo camera configuration was adopted.

3.3 Metric Localization

To estimate the exact position of the vehicle in metric localization approaches, static structures must be taken as references from environment for mapping and localization. The urban environments consist of many different static structures that could be used

as reference, but at the same time, there are several dynamic obstacles that can cause confusion on localization algorithms. Because of this, the scientific community has been developing approaches to deal with uncertainties and to localize the vehicle reliably.

Figure 22 – The five factors to be considered on a metric localization approach.

Source: author.

However, to elaborate a metric localization approach, five main factors must be taken in consideration: map features, map representation, matching approach, probabilistic filtering, and sensors. Figure 22 shows these factors in a visual diagram. The map features are the structures from the environment that are mapped and taken as reference during localization. After defining the map features, one must decide the map representation that stores them reliably. During localization, a matching approach is needed to make the correspondence between sensor data and the map, defining the most probable position of the vehicle. But, since noise is present on sensor measurements, it could cause ambiguities and uncertainties during localization process. For this, probabilistic filtering is commonly used to avoid wrong estimations (THRUN; BURGARD; FOX, 2006). Lastly, since there are many available sensors that can be used for mapping and localization purpose, it is important to know about the advantages and disadvantages of each sensor type for better choosing.

3.3.1 Map Features

Urban scenarios consist of a complex combination of static structures, so there are several possibilities of features for mapping. Some authors prefer to use every static structures as map features (WOLCOTT; EUSTICE, 2015; CASELITZ et al., 2016) with the intention of getting a robust geometric representation. Consequently, the localization approach would suffer less with noise caused by occlusions.

Besides, there are more efficient ways of representing the environment for localization purpose. According to Wolcott and Eustice (2014), using the whole scenario structure as map feature does not increase the localization performance as much as the computational burden needed to store such data. For instance, simpler and more representative features have been used, such as building facades (BAO et al., 2016; WEI; CAPPELLE; RUICHEK,

2013), road markings (SCHREIBER; KNÖPPEL; FRANKE, 2013; JO et al., 2015; HATA; WOLF, 2016; CUI; XUE; ZHENG, 2016; DU; TAN, 2016a), road boundaries (SCHREIBER; KNÖPPEL; FRANKE, 2013) and curbs (HATA; WOLF, 2016). In case of vision-based localization, features that are nearer the vehicle most of the time are preferred (e.g. road markings and curbs), as already reviewed in previous section.

3.3.2 Map Representation

In metric approaches, the localization performance depends on the map representation quality. For an accurate localization, a map must be sufficiently precise and have sufficient resolution to represent the geometric information of the environment (HATA, 2016).

One way to ensure sufficient data in the map for a reliable localization is by adopting a 3D representation of the entire environment, such as point clouds obtained by a 3D laser scanners (CADENA et al., 2016). But, a high detailed 3D map comes with high computation necessity for localization, and in large scale, the required memory would be a problem (BRUBAKER; GEIGER; URTASUN, 2016; BAO et al., 2016).

Alternatively, 2D representations have been widely explored. As acquiring a precise representation of the environment by mapping process is a laborious task (HATA, 2016), authors have been using maps provided by third parties, like the global information system (GIS) (WEI; CAPPELLE; RUICHEK, 2013; LIU et al., 2015) and the Open Street Map (OSM) (JO et al., 2015). However, the access of GIS maps is paid and it is not available for all locations. On the other hand, OSM is an open collaborative project that provides a free editable map of the world. But, since it is a crowd sourced data, its quality might not be as reliable as the GIS maps.

On the field of mobile robotics, the occupancy grid map (OGM) (MORAVEC, 1988; ELFES, 1989) is still a predominant paradigm in metric approaches, and is widely applied to outdoor environments due to its clarity on describing free and occupied spaces (THRUN; BURGARD; FOX, 2006; CADENA et al., 2016). Although this map type had been develop to represent the occupancy of spaces, recent works have been using variations of it. For example, in case of using LIDAR, road markings can be detected by the infrared reflectivity value and be assigned to a grid map (LEVINSON; MONTEMERLO; THRUN, 2007; HATA; WOLF, 2016). Note that the values assigned to each grid does not define the occupancy anymore, instead, they define the presence of road markings.

As grid maps are discrete representations of the environment, if not using enough resolution, it will affect directly the precision of the localization. To overcome this problem, Hata (2016) proposed the use of a continuous occupancy map generated by Gaussian Process, referred as Gaussian Process Occupancy Map (GPOM). Having a continuous representation, one can determine the occupancy of any point location, even those that

have not been observed previously.

3.3.3 Matching Approach

Once the features of the environment are represented on a map, a matching approach is needed to compute the correspondence between detected features and the map to define the vehicle localization. For instance, there are two approaches widely used in metric localization, namely Iterative Closest Point (ICP) and Likelihood Fields (HATA, 2016).

The ICP, as the name already says, is an iterative process that matches observed points with map points by minimizing the distances between them. In the method proposed by Segal, Haehnel and Thrun (2009), a generalized ICP is employed to match two point clouds obtained by 3D LIDAR. For localization purpose, this could be applied to match an observed point cloud with a 3D map. But, it would be impracticable due to high computation costs. So, recent works have been applying ICP to match sensor data with 2D maps instead (POMERLEAU; COLAS; SIEGWART, 2015). For example, Schreiber, Knöppel and Franke (2013) used ICP to match detected lane markings from image with the local grid map.

The other matching approach, Likelihood Fields, applies a probability distribution function on the observed points to deal with uncertainties. A common probability distribution function used is the Gaussian function, which provides smoothness on measurements, and deals better with ambiguities (THRUN; BURGARD; FOX, 2006). Furthermore, because the result of the matching process is given by a probability, it can combine more than one type of map features or sensor. For example, in (HATA; WOLF, 2016), two map features were taken (road markings and curbs) to match with an OGM during localization. Alternatively, Jo et al. (2015) proposed a system based on multi-cameras setup, and used individual measurement likelihood for each camera. Then, during localization, the final measurement probability was computed by the production of every individual likelihood.

In both techniques, the main drawback is the computational cost during the matching process. However, Likelihood Fields approach has an advantage over ICP in this issue. In practice, the likelihood of each map location can be pre-computed, simplifying the calculation of a measurement probability to a coordinate transformation followed by a table lookup (THRUN; BURGARD; FOX, 2006; HATA, 2016).

3.3.4 Probabilistic Filtering

In mobile robotics, the map-aided localization problem consists in localizing the robot by matching sensor data with the map. But for localization feasibility, there is a restrictive assumption that the environment is static. This assumption, called as Markov assumption, is violated when the robot is inserted in a dynamic environment, e.g. the case of autonomous vehicles in urban scenarios. However, such violation can be robustly handled

by Markov localization approaches, which are Bayes filters applied to the localization problem (FOX; BURGARD; THRUN, 1999).

Figure 23 – Description of Markov localization in diagram.

Source: author.

The key idea of Bayes filters is that a belief distribution is firstly predicted by a control model, then it is updated by a measurement model. When applying it in map-aided localization problem, the map is taken as an additional input data, and the robot motion model is taken as the control model (THRUN; BURGARD; FOX, 2006), as can be seen in the Figure 23. Currently, the most widely used Markov localization approaches for autonomous vehicles are based on the Extended Kalman Filter (EKF) and the Particle Filter.

The original Kalman Filter(KF) algorithm is designed for linear system models with Gaussian uncertainties. Since the pose of a vehicle is not governed by a linear function, the EKF is used instead. The EKF approach approximates the motion model of a vehicle to a linear model, by Taylor series expansion. Although it is just an approximation, EKF works well for most vehicles (THRUN, 2002). Due to its low computation burden and a simple calculation framework, some authors adopted this solution for Markov localization (SCHREIBER; KNÖPPEL; FRANKE, 2013; WOLCOTT; EUSTICE, 2014; GRUYER; BELAROUSSI; REVILLOUD, 2014; WOLCOTT; EUSTICE, 2015). However, EKF localization is usually limited to non-cluttered environments, and cannot handle ambiguous measurement caused by two or more similar features located in different positions (JO et al., 2015).

Different from EKF, that is a parametric and uni-modal approach, the Particle Filter is a non-parametric and multi-modal approach. Furthermore, it does not assume linear models, so it does not have limitations with nonlinear motions. In literature, the application of Particle Filters on localization problems is frequently referred as Monte

Carlo Localization (MCL). Because of its advantages over EKF and its simplicity to be implemented, MCL approaches have been widely used (LEVINSON; MONTEMERLO; THRUN, 2007; JO et al., 2015; XIAO et al., 2016; HATA; WOLF, 2016; DU; TAN, 2016a). Nonetheless, as any approach has its disadvantages, the main drawback of MCL approaches is the high computational cost in large particle sets.

3.3.5 Sensors

The two sensors that have mostly been applied for map-aided localization are LIDARs (e.g. Velodyne) and cameras (e.g. monocular, stereo camera and multi-cameras setup). Multilayer LIDARs are frequently preferred on the field of mapping and localization research due to their precision and independence on external lighting source (LEVINSON et al., 2011; WOLCOTT; EUSTICE, 2015; HATA; WOLF, 2016), and are said to be essential on autonomous vehicles (WEBB, 2017; SIMONITE, 2017). But, due to their sizes, high costs and mechanical operation, multilayer LIDARs are still a prohibitive solution for commercial autonomous vehicles (GARCIA-FIDALGO; ORTIZ, 2015; WEBB, 2017). However, some companies have announced smaller and cheaper solid-state LIDARs, which may favor the choice for this type of sensor (ROSS, 2017).

Nevertheless, for a while, cameras are much smaller, lighter and cheaper than LIDARs, not to mention the advantages in relation to the density of information that can be abstracted from images (e.g. colors, shapes, textures, and even depth). For this reason, the number of vision based solutions have been increasing on the past few years (FUENTES-PACHECO; RUIZ-ASCENCIO; RENDÓN-MANCHA, 2012; GARCIA-FIDALGO; ORTIZ, 2015), showing that it is possible to have a reliable and accurate performance without LIDAR sensor, as already discussed in previous section. Also, although cameras are subjected to lightning conditions, late approaches for road estimation and image segmentation based on deep learning have shown robustness against this factor (MENDES; FREMONT; WOLF, 2016; BADRINARAYANAN; KENDALL; CIPOLLA, 2017).

4 ROAD FEATURES DETECTION

This chapter describes the main contribution of this work, which is an approach to provide a tridimensional perception of the road environment through road features. Basically, it is a method to detect road features on monocular images by a FCNN named as 3CSeg-Multinet, and project them to a point cloud by stereo vision. The following sections are going to give a detailed explanation of the FCNN architecture used for road features detection (in Section 4.1) and the method for point cloud projection (in Section 4.2), as well as their motivations and methodologies for implementation.

4.1 The 3CSeg-Multinet Architecture

The architecture employed for road features detection on images was only the part of the MultiNet (TEICHMANN et al., 2016) used for image segmentation, which was adapted for road features detection. This architecture can be seen in Fig. 24. Since it is a FCNN, the input to the network can be a three-channel monocular image of any spatial size, and the output is an array of the same spatial size of the image, but with three channels corresponding to each class (road marking, curb and background). The encoder correspond to the 13 first convolutional layers (with 5 max pooling) of the VGG16 network (SIMONYAN; ZISSERMAN, 2014), and three convolutional layers instead of fully connected layers to provide low resolution segmentation. For decoder, three deconvolutional layers are employed as in (LONG; SHELHAMER; DARRELL, 2015). Lastly, for better optimization of such deep network, two residual connections (HE et al., 2015) are used.

On the encoder part, the architecture is divided in five blocks, where every convolutional layers use 3×3 kernel with stride 1 and *half padding* (to keep the spatial size of the feature maps), and all the max pooling layers have 2×2 receptive field with stride 2 (to reduce the resolution by half). As the depth increases, the number of filter increases and the spatial size (or resolution) decreases, creating the *bipyramidal effect*, as discussed in Section 2.1. In this way, the network starts with 64 filters on the first block and doubles it for each further block, ending with 512 filters. Therefore, the encoder outputs 512 feature maps with resolution reduced by a factor of 32, where each cell can abstract information of a 212×212 region from the input image (according to the Equation 2.3).

Next, three convolutional layers are employed to produce low resolution segmentation as output. The first layer consists of 4096 filters of 7×7 kernel, which allows to capture contextual information of a 404×404 region from the input image. The following layer also applies 4096 filters, but the convolution uses 1×1 kernel. Lastly, to have the low resolution segmentation, the third layer consists of 3 filters of 1×1 convolution. In this part, every layer applies unit stride and half padding.

Figure 24 – Diagram with detailed network architecture. The input RGB image can be of any *size*, and the output is given by a two channel array of the same *size*.

Source: author.

Finally, the decoder is formed by three deconvolutional layers, where the first two layers upsample the feature maps by a factor of 2, while the last layer upsamples by a factor of 8. For this, the first two layers employ deconvolution with 4×4 kernel and stride 2. And the third layer applies a 16×16 deconvolution with stride 8. For every deconvolutional layers it is used half padding.

The plain architecture employed in this work is quite deep (16 convolutional layers, 5 max pooling, and three deconvolutional layers), so it would be prone to gradient vanishing during training process (TEICHMANN et al., 2016). To deal with this, two residual connections of 1×1 convolution (HE et al., 2015) are used: one between the

fourth max pooling layer and the second deconvolutional layer, and other between the third max pooling layer and the third deconvolutional layer.

4.2 Features Projection to Point Cloud

On the previous section, it was presented the main contribution of this chapter, which is an approach based on a deep FCNN to detect road features on monocular images. However, detecting road features on images still does not give the tridimensional perception for the vehicle. To deal with it, a stereo camera can be used to provide depth information and project the road features to the 3D space (refer to Section 2.3).

To recover only the 3D points correspondent to the road features, it is necessary to follow four simple steps, namely road features detection, disparity computation, disparity filtering, and 3D reconstruction. These steps are illustrated in Figure 25.

Figure 25 – Illustration of the steps for road features projection to a point cloud.

Source: author.

First, the proposed road features detector is executed on the left image only, and the *correlation-based correspondence* (refer to Section 2.3.1) estimates the dense disparity map. Then, the detected features on the left image is used as a mask to filter the disparity map, extracting only the disparities of the pixels correspondent to the road features. Finally, those disparities are used to project the road features in a point cloud by 3D reconstruction described in Section 2.3.2.2.

In this way, the vehicle would be able to perceive in low level the space of urban roads. As mentioned in Chapter 1, in order to check if the proposed visual perception system indeed is applicable to autonomous vehicles, this work also proposes a visual-based metric localization model based on MCL (refer to Section 2.5). This localization model is going to be detailed in the next Chapter.

5 THE LOCALIZATION MODEL

In this chapter, it is proposed a vision-based localization model as an application for the road features detector described in Chapter 4. The employed localization model is basically the MCL model with KLD-sampling explained in Section 2.5, including an OGM as the map provided for localization, an initialization by a low-cost GPS, motion model based on odometry estimated by a low-cost GPS and an IMU, and the proposed road features detector adapted for measurement model. Figure 26 shows a block diagram of the proposed localization model.

Figure 26 – Diagram of the proposed localization model.

Source: author.

Because LIDAR provides more precise range information than a stereo camera, it was chosen to be the sensor for mapping. So, the OGM for this localization model is a 2D map generated by a multilayer LIDAR, according to the method proposed by Hata and Wolf (2016). In this mapping approach, curbs and road markings are used as map features. Curbs are detected by their step shape, and road markings by the reflective intensity of returning infra-red beam.

Once a map is given, it is taken the average values of N measurements of position and orientation provided by a low-cost GPS and the magnetometer of an IMU respectively for localization initialization. Then, still with these sensors, the odometry is estimated as the motion model for a priori belief estimation. The translation motion is given by the position derivative from GPS data, and the orientation is estimated by the IMU's

magnetometer values and angular velocities. Finally, the proposed approach for road features detection is employed for measurement model. But, to enable fast matching between the detected road features point cloud and the 2D OGM by Likelihood Fields (refer to Section 2.4.1), it is used an approximation to simulate a laser scanner sensor over the point cloud. The following section gives a more detailed explanation about this adaptation for measurement model.

5.1 The Measurement Model

In Chapter 3 it was discussed about two matching approaches widely used for finding correspondences between the detected features and the map, which are ICP and the Likelihood Fields. Because the proposed method for road marking detection in 3D space gives a point cloud as result, it is natural to think about using ICP. But, running the ICP algorithm for a dense point cloud and the map may be costly. Aiming to get a real-time performance, it is advantageous to employ Likelihood Fields instead, since it can be pre-computed before localization.

However, Likelihood Fields is a range sensor model, making it difficult to apply in a point cloud. For this, one can use an adaptation that provides a way to convert a point cloud partially to a laser scan type measurement. In this way, it is possible to use Likelihood Fields to match the converted measurements with the map. This adaption is provided by the *pointcloud_to_laserscan*¹ ROS package, which is open sourced under a software license agreement (BSD License).

Figure 27 – Point cloud to laser scan conversion model.

Source: author.

Basically, it simulates a laser scan that searches for obstacles in a region defined by a range of distance ($d_{min} < d < d_{max}$), a range of height ($h_{min} < h < h_{max}$), and a range of angle ($\theta_l < \theta < \theta_r$). Figure 27 shows a simple illustration of this search region. Thus, it starts from θ_l to θ_r , scanning for each step s . But, the obstacles that it searches for

¹ Source: BOVBEL, P. **pointcloud_to_laserscan - ROS Wiki**. 2015. Available from Internet: <http://wiki.ros.org/pointcloud_to_laserscan>. Accessed in: jan 2018.

are points of the point cloud. Because it considers each point as an obstacle, it cannot detect a point occluded by another point and, consequently, it cannot convert the entire point cloud to a laser scan type measurement. Although this simulated laser scan searches for points in 3D space, it takes only the angle and distance where a point was detected, resulting in a 2D range measurement.

From Figure 28 it is possible to see a clear example of how the partial conversion from point cloud and laser scan measurement works. The points highlighted with red contour are the detected obstacles, while the black points with a blue dashed line are the ones not detected due to occlusion or location out of the search region.

Figure 28 – An example of conversion from point cloud to laser scan measurement.

Source: author.

In summary, the adopted localization model gets low-cost GPS data as initialization, and an odometry estimation as the motion model for the prediction step, and finally, as the main contribution, the proposed visual perception system is applied in the measurement model for the updating step. Briefly, the measurement model consists in detecting the road features on the left images of the stereo camera, projecting them to a dense 3D point cloud through estimated disparity, applying the point cloud to laser scan conversion, and executing the Likelihood Fields for matching with the map.

6 EXPERIMENTS AND RESULTS

This chapter describes in three sections the experimental methodology, framework and results of road features detection in images, 3D projection of road features, and the test of vision-based localization in a map generated by LIDAR. The first section details the methodology adopted to train, validate and test the FCNN employed for road features detection, as well as the results. Then, the second section presents the experimental framework and the results of 3D projection of the detected road features. Lastly, the third section explains how localization test was done, and discuss about the achieved results.

All the experiments were executed offline. Therefore, datasets with all necessary sensor data had to be collected. It was done with the CaRINA II platform (SHINZATO et al., 2016), an ongoing autonomous vehicle project held by the INCT-SEC (Brazilian National Institute of Science and Technology for Critical Embedded Systems) and CRob-SC (Center for Robotics of University of São Paulo at São Carlos).

6.1 Road Features Detection

This section consists of four subsections, namely training scheme, experimental framework, and results. First, in the two first subsections, it is detailed the methodology adopted for training the FCNN architecture, and the experimental framework (including the issue about dataset, which had to be manually generated). Then, the last subsection presents the results, discussing about the strengths and weaknesses of the proposed approach for road features detection.

6.1.1 Training Scheme

To classify every pixel on the image, it was used pixel-wise one-hot encoding. Therefore, each pixel has three output values corresponding to each class (road marking, curb and background). For training, the cross-entropy function was employed as the loss function, which is defined as

$$loss_{pixelwise}(y_o, y_{gt}) = -\frac{1}{|I|} \sum_{i \in I} \sum_{c=1}^3 y_{gt}(c) \log y_o(c), \quad (6.1)$$

where, y_o , y_{gt} , c and $|I|$ correspond to network output, ground truth, class and size of the mini batch used for training, respectively.

Then, to minimize the loss function, Adam optimizer (KINGMA; BA, 2014) was used as the stochastic optimization algorithm. For this work it was adopted the following hyper-parameters: learning rate of 1×10^{-5} , first moment decay of 0.9, second moment decay of 0.999, and the stability numerical term of 1×10^{-5} . Also, to learn more robust

contextual features, a dropout with 50% of probability was employed for the two last layers of the low-resolution segmentation part.

Furthermore, aiming to get better optimization, the parameters were initialized with pre-conditioned values. For each part of the network architecture, a different initialization was done. On encoder and low-resolution segmentation parts, the parameters were initialized with VGG16 weights, which were pre-trained with ImageNet Dataset and provided by Simonyan and Zisserman (2014). For decoder part, as it actually upsamples the segmentation maps, bilinear upsampling kernels were employed for initialization. Lastly, residual connections were the only parts initialized with random values. The residual connection between the fourth max pooling layer and the second deconvolutional layer was initialized with random values around $\pm 1 \times 10^{-3}$, while the other residual connection, with values around $\pm 1 \times 10^{-4}$.

Regarding the evaluation, it was done by computing the rate of correctly inferred pixels according to the ground truth, which is called accuracy in this work. The accuracy is defined by the following equation:

$$Accuracy = \frac{TP + TN}{N}, \quad (6.2)$$

where, TP , TN and N correspond to the number of pixels considered as true positives, true negatives and the total number of pixels on the image, respectively.

6.1.2 Experimental Framework

6.1.2.1 Dataset

The FCNN architecture employed for this work is quite deep and complex, which means that it has a large VC-dimension (Vapnik-Chervonenkis dimension) dimension. Because of this, a large dataset would be necessary to avoid overfitting (LUXBURG; SCHOELKOPF, 2008). However, there was no available labeled dataset for road marking and curb detection. Therefore, a new dataset had to be generated and manually labeled.

Initially, 216 images of the CaRINA dataset (SHINZATO et al., 2016) were used, consisting of road scenes from the campus I of USP-SC. From these images, 181 were used for training and 35 for validation. Later, aiming to vary the scenario, it was used 124 more images previously collected with the CaRINA II platform in the campus II of USP-SC. Thus, in total, 340 images were gathered for training and validation. More specifically, 300 images for training, and 40 for validation. Furthermore, to get better results and decrease the probability of overfitting, it was adopted the data augmentation technique, which consists in doubling the dataset size by flipping each image horizontally, and cropping a region of 404×404 at a random position of each image for each iteration. Regarding the test, unfortunately no more labeled dataset could be created.

The size of each image on this new labeled dataset is 1280×960 . Since road features are present just on the lower part of the image, a region of interest (ROI) was defined as being the 490 lowest rows of pixels. Figure 29 shows a sample of CaRINA dataset with the defined ROI, and its ground truth.

Figure 29 – A sample from CaRINA dataset (SHINZATO et al., 2016) and from additional collected images. (a and c) Original images with ROI defined in red, (b and d) ground truth.

Source: author.

6.1.2.2 Setup

The images were collected with the Pointgrey Bumblebee XB3, which is a stereo camera. For road marking detection, only images obtained by the left camera were used and manually labeled with the Gimp software.

Regarding the FCNN implementation, it was done using Python language and TensorFlow-1.0 framework (ABADI et al., 2016), and trained in a machine with Intel Core i7-4960X and 16GB RAM. Later, an NVIDIA Titan X Pascal with 12 GB of memory was donated to USP-SC by NVIDIA Corporation, under a research equipment grant. So,

the tests could be executed in a machine with Intel Core i7-7700K, 16GB RAM, and this GPU, achieving an execution rate of around 10 fps.

6.1.3 Results

As already mentioned in Section 6.1.2.1, the dataset was manually labeled in two parts. On the first part, the FCNN network was trained with 181 images (and data augmentation technique), and validated with 35 images, resulting in here denominated as the *first classifier*. In Figures 30(a-b) is possible to see that there was a fast convergence in the validation set. Two possible reasons for this are the small training set and good parameters initialization. The training set used for such deep FCNN architecture may, very probably, lead to a very fast convergence. Moreover, considering that the encoder and decoder were initialized with well pre-conditioned parameters, and that only the residual connections were randomly initialized, it is expected to have less problem with stagnation during training. As result, the network has achieved over 99% of accuracy and under 0.01 of loss before 4000 iterations. After that, no more significant improvements has occurred, ending the 11900-th iteration with 99.58% of accuracy and 0.0095 of loss.

Another important observation is about overfitting. In Statistical Learning Theory (SLT), it is known that this phenomenon happens when a machine learning model is trained too much with a single biased dataset. Theoretically, it can be observed on the loss

Figure 30 – Convergence of the first training process.

(a) Loss (cross-entropy).

(b) Accuracy over the validation set.

Source: author.

curve obtained through inference on the validation set. If the loss starts to grow after a certain quantity of training iterations, it means that the machine learning model is starting to "memorize" the training data. However, it could not be observed on the accuracy and loss curves in Figure 30. From this, it is possible to get two hypothesis associated to each other. First, the images used for validation are very similar to those used for training. Secondly, because of this similarity, it is very probable that overfitting has occurred.

Figure 31 – Images that the FCNN fails to detect road features correctly.

Source: author.

Since the images used for training and validation consists of a sequence of frames obtained in one scenario only, then the first hypothesis is true. To check about overfitting, few tests were done with different scenarios. However, there was no more labeled data, so the evaluation was done visually. Figure 31 shows two images obtained in the campus II of USP-SC, where the scenario consists of larger roads and less occlusions if compared to the dataset used for training. Apparently, the environment is less cluttered, but it is possible to see that the *first classifier* could not detect all the road markings (in green) and curbs (in red), and at the same time, there were some false detections. Therefore, the hypothesis that overfitting has very likely happened is also correct.

Figure 32 – Convergence of the second training process.

Source: author.

Aiming to get better results, 124 more images previously collected were added to

the training and validation sets. Then, the FCNN was trained again, but starting with the parameters trained before. Because of this, it is expected faster convergence than in the first training process. In Figures 32(a-b) it is shown that the training converged within 2000 iterations, reaching 99.07% of accuracy and 0.0432 of loss. Some tests have shown better detection performance, but it still seems to have the overfitting problem. The retrained FCNN network is here denominated as *second classifier*.

Figure 33 shows the difference of detection performance between the network inferences after the first (the left column of images) and second (the right column of images) training, where red highlights indicate curb detection and green the road markings. The first two images are scenes from the campus I of USP-SC, so it was expected to have good performance in both cases, being able to detect the road features even in regions with some shadows and occlusions. But, the four next images were taken from the campus II of USP-SC, and it is clear that the detection performance had improved after training the network with additional images. On the third image, the retrained FCNN has shown robustness even against light reflex on road surface. However, it is still possible to see some wrong detections in the last two examples. For example, zebra-crossing could not be detected correctly in the fifth image, and a part of the central road marking was detected as curb in the last image. Also, in both last images, some curbs could not be detected due to strong tree shadows.

Due to the lack of labeled dataset and different evaluation protocols it is hard to make a quantitative comparison between the proposed method and other methods. However, as discussed in Chapter 3, most of algorithms for road marking detection would detect just lane markings in regions without intersections only, and additional techniques would be necessary for each different road marking pattern, not to mention curbs. But, experimental results showed that the proposed method could detect most of road markings and curbs in just a single forward inference. This is because deep architecture of consecutive convolutional layers, on the encoder part, allows the network to obtain contextual information over a large region around the pixel to be classified, unlike approaches based on local gradient that uses only neighboring pixels information. In this way, the network outputs a smoother segmented image.

Nonetheless, there are three main observations that must be considered. First, it is still challenging to deal with images cluttered with strong shadows and degraded road marking or curb, which makes it hard to detect even with human eyes. Second, the robustness against occlusion mentioned before consists in not classifying objects that cause occlusions as road features. Therefore, the proposed approach cannot detect occluded road features. Lastly, the size of labeled dataset used for training and validation may still be insufficient for generalization guarantee after training. Having a deep FCNN architecture (in this case, with more than 134 million parameters) with high VC-dimension and small

dataset makes the network prone to overfitting (LUXBURG; SCHOELKOPF, 2008). To get better results, using bigger dataset or simpler architecture would be necessary.

Figure 33 – Random images used for test. The left column of images shows the results obtained after the first training, while the right column shows the results obtained after the second training.

Source: author.

6.2 3D Projection of Road Features

The next step after getting results for road features detection on images is projecting the detections to 3D space as point cloud. This section is divided in two subsections, namely experimental framework and results. In the first subsection, it is briefly presented the tools (hardware and software) used for experiments, and how they were used. In results subsection, it is provided visual results and a detailed qualitative analysis.

6.2.1 Experimental Framework

As proposed in Section 4.2, the detected road features can be projected to a point cloud, which would be given in the stereo camera coordinate system (as explained in Section 2.3.2.2). But, if applied for a facility in an autonomous vehicles (e.g. vision-based localization), the point cloud must be in the coordinate system of the vehicle (as explained in Section 2.4.1). Therefore, a calibration was necessary to get the relative pose of the stereo camera to the vehicle's frame.

Figure 34 – Stereo camera and LIDAR installation on CaRINA II platform.

Source: author.

To collect data for experiments, it was adopted the stereo camera Pointgrey Bumblebee XB3. But, before data collection, it was used a Velodyne HDL-32E LIDAR to adjust the stereo camera pose in relation to the vehicle only. Regarding the calibration of Bumblebee XB3, it is provided by its manufacturer. Both sensors were installed on the CaRINA II platform, which counts with predefined positions for their installation, as shown in Figure 34. However, for each time the sensors are installed, they may have small variations on their orientations. Nonetheless, once the LIDAR orientation is adjusted according to the vehicle's frame, it is possible to set the calibration between the stereo camera and LIDAR that aligns a set of points from the point cloud with their correspondent pixels on the stereo image, which can be done empirically. Since LIDAR is already calibrated with the vehicle's frame, it is possible to get the calibration between the stereo camera

and the vehicle. Figure 35 shows the sensors calibration with the vehicle (the vehicle's coordinate system is named as *base_link*).

Regarding the implementation of the *correlation-based correspondence* algorithm and the 3D projection, it was done in Python and OpenCV-3.0.0 package. Also, to visualize point clouds and if they were in accordance with the vehicle's coordinate system, it was used the RVIZ visualization tool provided by ROS (Robot Operating System) and the simulated model of CaRINA II developed by Shinzato et al. (2016).

6.2.2 Results

There was no ground-truth to compute the precision of the road features projection, so a visual comparison was done with the point cloud provided by LIDAR. For this, stereo images and point cloud from LIDAR were collected simultaneously in a straight road from campus II of USP-SC. Figure 36(a) shows the detected road features (highlighted in green), and Figures 36(b-c) show different views of the point clouds (from LIDAR and from road features projection). In Figure 36(b), it is possible to observe qualitatively the vertical accuracy of the road features projection. Although the left curb and central lane markings seems to be aligned with the road surface defined by the LIDAR's point cloud, the projected right curb shows a horizontal offset from the curb defined by LIDAR, and it presents a vertical decay as it gets farther from the vehicle. With a bird's-eye view perspective in Figure 36(c), it is clearer the horizontal offset of the right curb. However,

Figure 35 – Calibration between sensors (*Velodyne* and *Bumblebee*) and the vehicle's frame (*base_link*) displayed in RVIZ. In this illustration, the vehicle's frame is denominated *base_link*, and *base_footprint* is the coordinate system of the vehicle in ground level. The yellow arrows represent the translation and rotation transformations between two coordinate systems.

Source: author.

the projected features that bound the lane where the vehicle is located (in this example, the right curb and central lane markings) seems to be more precise.

Figure 36 – Comparison between point cloud given by LIDAR (colorful 32 concentric rings of points) and the road features point cloud (in white).

Source: author.

In theory, the accuracy and precision of a point projection from image to 3D space should be better for regions near the stereo camera. As seen in the above example, precision is indeed better for projections near the stereo camera, but it is not the same for accuracy. The error of the right curb projection is visible, and bad camera calibration may be a cause. Regarding the imprecision for larger distances, it is inevitable due to discrete nature of the digital images. So, as an object moves away from the stereo camera its disparity on images gets smaller, suffering with numerical approximations.

Aiming to verify the proposed approach of road features detection and projection in several conditions, a dataset consisted of a sequence of frames was collected in campus II of USP-SC. Unfortunately, because of memory limitation, LIDAR data could not be included in this dataset. The test was executed in the machine with NVIDIA Titan X Pascal, running at 10 fps. Figure 37 shows some results of road features projection in situations where the proposed approach showed to be robust against shadows (a), dust on road surface (b), lightning effects (c), and it did not misclassified obstacles on roads as road features (d). On these cases, it was possible to have a low-level perception of the road environment. In this way, it may be used to define if the vehicle is in a straight or curved road, as well as if the vehicle is changing the lane (Figure 37(f)).

However, there are some challenging issues that were not solved with the proposed

Figure 37 – Some good results for road features detection and projection. On images, the detected road markings are highlighted in green, while curbs are in blue.

Source: author.

approach yet. Since only pixels classified as curb or road marking are projected to 3D space, road features must be accurately detected in every conditions. But, the proposed approach still cannot detect degraded road markings and curbs in shadowed area (Figure 38), which jeopardizes the spatial perception of the vehicle. It may be associated with two possible reasons. First, the overfitting problem already discussed in previous section

(6.1). Second, regions of the image with low contrast caused by shadows makes it harder to recognize patterns.

Figure 38 – Some bad results for road features detection and projection. On images, the detected road markings are highlighted in green, while curbs are in blue.

Source: author.

The last unsolved issue to be considered is about tilted projections. Since the stereo camera is installed on a hard support over the vehicle, if the vehicle pass by any non-planar surface, the road features projection will be tilted. Figure 39 shows an example in which the vehicle moves through a speed bump. In this example, the projection varies only the pitching angle, but there may be some cases of rolling angels too. If the proposed approach is applied for a task that considers roads as planar surfaces, those angular variations may impair the accuracy of the system.

Figure 39 – Road features projection tilted because of a speed bump.

Source: author.

6.3 Vision-Based Localization

The previous section showed that if the road features detection approach is robust enough, it can be projected to 3D space in order to have better spatial perception of the road environment. This could be used as a tool for some facilities of an autonomous vehicle, e.g. lane keeping system and localization. To validate the applicability of the proposed method, it was tested as the measurement model of the MCL localization algorithm for an urban scenario.

6.3.1 Experimental Framework

The localization test was done offline, so all necessary data had been previously collected with the CaRINA II platform, which had run a trajectory of around 1.5 *km* in campus II of USP-SC (two rounds on the block shown in Figure 40(a)) at a speed up to around 35 *km/h*. The sensors used for this experiment were as follows:

- Stereo camera: Pointgrey Bumblebee XB3.
- Low cost GPS: Garmin 18x.
- IMU: Xsens MTi-300.
- High precision GNSS: Septentrio AsteRx2eH PRO with Real Time Kinematic (RTK).

The three first sensors were employed to the proposed localization model (refer to Section 5), and the high precision GNSS was used to get the ground truth data.

Figure 40 – Maps of the place where localization experiment was executed.

Source: (a) Satellite image taken and edited from Google Maps (2018); (b) OGM provided by Hata and Wolf (2016); (c) OGM divided in 9 regions, where red regions correspond to curves or roundabouts, and blue or green regions correspond to straight roads.

Regarding the OGM generated by a multilayer LIDAR (Velodyne HDL-32E), it was kindly provided by Hata and Wolf (2016). This map can be seen in Figure 40(b), consisting of 1999×3069 cells with 0.1 m of resolution. Aiming to evaluate the localization performance of the proposed localization model, the map was divided in 9 regions, as shown in Figure 40(c).

Lastly, to implement the localization model, two ROS packages were used, namely *pointcloud_to_laserscan*¹ and *amcl*². The first package was used to convert projected point clouds in laser scans (refer to Section 5.1), and the second to execute the MCL algorithm with KLD-sampling (refer to Section 2.5).

6.3.2 Results

For this experiment, the point cloud to laser scan conversion (refer to Section 5.1) was employed and a plane environment was assumed. Since there were speed bumps on the trajectory where localization was tested, the issue about projection tilting (discussed in the last paragraph of Section 6.2) had to be considered to define the height range of the laser scan search region. Also, the horizontal field of view of the cameras (used by the Pointgrey Bumblebee XB3) was taken into account to define the angle range, and the precision of depth estimation by stereo vision to define the range of distance. Therefore, the search region was constrained by a distance range of $0.5 < d < 10\text{ m}$ from the vehicle's front, $-3.0 < h < 3.0\text{ m}$ of height range, and $-33^\circ < \theta < 33^\circ$ of angle range.

Figure 41 – Example of the point cloud to laser scan conversion in case of speed bump. Point clouds are represented in green, and measured points by laser scan in red.

Source: author.

This conversion model is an approximation, and for tilted point cloud projections produced by irregularities on the road surface may cause inaccuracies. However, the largest

¹ Source: BOVBEL, P. **pointcloud_to_laserscan** - ROS Wiki. 2015. Available from Internet: <http://wiki.ros.org/pointcloud_to_laserscan>. Accessed in: jan 2018.

² Source: SHAH, V. **amcl** - ROS Wiki. 2017. Available from Internet: <<http://wiki.ros.org/amcl>>. Accessed in: jan 2018.

tilt that occurred on this experiment was caused by a speed bump, and it was less than 10° of pitching angle. Thus, the largest inaccuracy in this experiment may have been the decrease of measured distance by a factor of less than $1 - \cos(10^\circ)$, which is approximately 0.015. Figure 41(a-b) show examples of laser scan conversion in cases of the vehicle moving up and down a speed bump.

Another important observation is about which road features points were matched with the map by Likelihood Fields. Because the conversion model sees each projected point as an obstacle, only the nearest visible points are measured and matched with the map. So, even though the proposed approach can detect and project a dense point cloud of road features, it won't be totally used in this localization method, as it is shown in Figure 42. However, although it doesn't use all information provided by the point cloud, it becomes computationally less expensive to match with the map.

Figure 42 – Some results of the point cloud to laser scan conversion. Point clouds are represented in green, and measured points by laser scan in red.

Source: author.

In localization experiment, MCL was set to use 100 to 500 particles. As result, the proposed localization model presented an absolute average error (Euclidean distance to ground truth) of $2.0042 \pm 1.2252 \text{ m}$, against an $2.3370 \pm 0.8576 \text{ m}$ from the low-cost GPS. Thus, for position estimation, the proposed method showed to be more accurate and less precise than a low-cost GPS. Since the motion model employed for MCL algorithm was based on the odometry estimated through the derivative of consecutive GPS data, the *priori* beliefs have very likely inherited imprecisions from the sensor and affected the performance of the localization estimations. Regarding the orientation estimation, the proposed method allowed to have an overall error of $0.0463 \pm 0.0308 \text{ rad}$, against $0.0575 \pm 0.0337 \text{ rad}$ given by an IMU. So, in this case, the proposed localization model showed to be more accurate and precise than the sensor by itself. Nonetheless, the overall results are not much better than raw sensor data. But, overall errors omit the performance of the localization model for each region of the map. So, a more detailed evaluation was

done to check for situations that the proposed localization model works better, and where are its weak points.

For better understanding the performance, the trajectory was divided in 9 different regions (see Figure 40(c)), and for each region a sample of estimations was taken to calculate longitudinal (x), lateral (y), Euclidean (xy) and orientation (θ) errors. Two error calculations were adopted, namely mean error and absolute average error. Mean errors allow to check where most of pose estimations were in relation to the ground truth, i.e. verify whether the pose estimations were mostly in front or behind, on the right or left side of the ground truth, as well as if their orientations were mostly to the east or west of the ground truth. About the absolute average errors, they provide the accuracy and precision performances.

Table 4 – Mean errors for each region of the map.

Region	x (m)	y (m)	θ (rad)
1	-0.5398 ± 0.7650	-0.4973 ± 0.1032	0.0044 ± 0.0120
2	-1.8878 ± 0.7715	-0.5772 ± 0.1172	0.0107 ± 0.0253
3	0.9195 ± 2.4088	-0.7279 ± 0.3904	0.0183 ± 0.0446
4	-1.3703 ± 1.1030	-0.5538 ± 0.1086	0.0208 ± 0.0205
5	-0.6836 ± 1.3565	-0.5296 ± 0.1124	0.0068 ± 0.0159
6	1.0462 ± 0.8221	-0.5736 ± 0.5266	0.0165 ± 0.0294
7	-0.9968 ± 0.9911	-0.5170 ± 0.1284	0.0279 ± 0.0224
8	-0.9597 ± 1.0910	-0.3575 ± 0.0825	0.0499 ± 0.0180
9	-1.0703 ± 1.0537	-1.3667 ± 1.2109	0.0639 ± 0.0454

Source: author.

Table 5 – Absolute average errors for each region of the map.

Region	x (m)	y (m)	xy (m)	θ (rad)
1	0.9363 ± 0.5578	0.5079 ± 0.1038	1.0652 ± 0.4460	0.0128 ± 0.0081
2	2.0393 ± 0.7862	0.5890 ± 0.1178	2.1227 ± 0.7546	0.0275 ± 0.0130
3	2.5783 ± 1.2061	0.8260 ± 0.4025	2.7073 ± 1.2254	0.0483 ± 0.0244
4	1.7590 ± 1.0376	0.5644 ± 0.1092	1.8473 ± 0.9401	0.0293 ± 0.0194
5	1.5191 ± 0.8126	0.5414 ± 0.1130	1.6127 ± 0.7444	0.0173 ± 0.0100
6	1.3301 ± 0.7748	0.7787 ± 0.5387	1.5416 ± 0.6369	0.0337 ± 0.0198
7	1.4056 ± 0.9568	0.5327 ± 0.1294	1.5032 ± 0.8379	0.0358 ± 0.0228
8	1.4530 ± 0.8266	0.3669 ± 0.0830	1.4986 ± 0.7767	0.0530 ± 0.0184
9	1.5020 ± 0.9149	1.8260 ± 1.2951	2.3644 ± 1.4431	0.0784 ± 0.0417

Source: author.

Table 4 shows the mean errors for each region of the map. It is possible to see that the longitudinal mean error (x) was mostly negative, which means the estimations were often behind the ground truth. Also, it shows negative lateral mean error (y) for the whole

trajectory, indicating that practically every estimation was offset to the right side of the ground truth. This offset implies that the localization model has a lateral bias, which may be associated with the inaccuracy of the road features projection on the right side, which was discussed in the subsection 6.2.2. Both longitudinal and latitudinal mean errors have shown to be systematic, indicating that there is something wrong with the data handling after the road features are detected. Lastly, the estimated orientation showed a paltry mean error.

In relation to the absolute average errors, they are presented in Table 5. The longitudinal errors reveal large inaccuracy and variance for every region of the map (not less than 0.9363 ± 0.5578), which may be associated with uncertainties caused by ambiguous road features, as shown in Figure 43(b) and with some road marking faults on the map (Figure 43(a)). In contrast, lateral errors show more accuracy and precision, specially in regions with straight roads (or with soft curves). However, regions where road features

Figure 43 – Examples of longitudinal imprecision during localization. Particles are represented in small red arrows, the estimated pose is in green, and the ground truth in blue.

Source: author.

are not precisely mapped and the laser scan conversion does not take enough information cause confusion during localization. Those regions happened to be roundabouts and curves (regions 3, 6, and 9). For example, in region 9 showed by Figure 43(d), there should be a zebra-crossing and horizontal stop sign, but neither the map nor the measurements given by the laser scan conversion provide enough data for a robust vision-based localization.

Hata and Wolf (2016) have shown it is possible to have an accurate localization (longitudinal, lateral and angular absolute average errors of 0.1395 *m*, 0.2040 *m* and 0.019 *rad*, respectively) in this map by using a multilayer LIDAR (Velodyne HDL-32E), which covers 360° perception around the vehicle. Nevertheless, this work shows it is more challenging to have an accurate localization by vision-based perception in a map generated by LIDAR. Still, there are situations that proves it is possible to have accuracy, as shown in Figure 43(c). Furthermore, as seen in related works chapter (refer to section 3.2), Wolcott and Eustice (2014) and Schreiber, Knöppel and Franke (2013) have also shown it is possible to have an accurate and precise visual localization in maps generated by LIDAR. However, in these both cases, the maps had more information (3D information provided by a multilayer LIDAR, or more precise and complete features mapped with LIDAR, high resolution camera and manual refinement). Therefore, to improve the performance of the proposed localization model, more information from the point cloud of the projected road features may be used, and the map may be more precise and have less mapping faults. But, for a while, experiments have revealed consistent results, even for limited information from the projected point cloud, which indicates the applicability of the proposed approach for tridimensional road features perception in metric localization problem.

7 CONCLUSION

The high rate of traffic accidents has been urging the necessity for new technologies to promote safety on roads, like ADAS and self-driving cars. Among the systems that may comprise these new technologies, the one responsible for perceiving the road environment in the urban scenario is of utmost importance. For instance, curbs and road markings are the most salient features of roads that can provide this perception. However, most of the vision-based methods already published have focused on detecting only one type of feature (curb or a road marking with specific pattern) at once, or have adopted multiple techniques to detect both road features (refer to discussed related works in Chapter 3).

Therefore, it was proposed an approach based on a deep FCNN architecture of encoder-decoder type, which can make pixel-wise multiclass classification of the image in a single inference. The architecture used for this purpose was an adaption of the MultiNet (TEICHMANN et al., 2016) part responsible for image segmentation adapted for three classes (curb, road marking, and background), being denominated in this work as 3CSeg-Multinet. The deep encoder of this network provides contextual information of a large receptive field from the input image, allowing each cell of the low-segmentation part of the architecture to receive information from a 404×404 region of the input image.

For experiments, there was no labeled dataset available for road features detection benchmark, so a new one had to be manually generated. But, it was labeled only enough data to train and validate the network. Experimental results have shown that the adopted network can converge in a relative few training iterations if it is initialized with well-conditioned parameters, and can run in real time (around 10 fps) in a GPU (NVIDIA Titan X Pascal with 12 GB of memory). Moreover, in a visual analysis during tests, the proposed approach has shown efficiency and robustness against most of the clutter caused by several lighting conditions, shadows and the presence of obstacles. Notwithstanding, overfitting has very probably occurred during training due to a relatively small dataset compared to the complexity of the FCNN architecture used. Unfortunately, because there is not a standard protocol and dataset to evaluate road features detection, it was not possible to compare the results with existing methods in the literature.

Road features were designed to provide the spatial perception of the environment, but detecting them on monocular images is not enough for having this perception. Thus, it was also proposed to project the detected features to a point cloud through 3D reconstruction algorithm of stereo vision. By means of visual comparison with a point cloud given by a multilayer LIDAR data, the projected point cloud of the detected road features has shown to be coherent with the real scenario, but not as precise as LIDAR data though.

Lastly, with the purpose of verifying the applicability of the proposed approach for perceiving the road environment through curbs and road markings, it was proposed a vision-based metric localization model based on the MCL algorithm with Likelihood Fields as the measurement model. For this to be possible, it was used an adaption to downsample the projected road features point cloud and make them suitable for Likelihood Fields. For the experiment, a low-cost GPS and an IMU were used to provide the initial pose and the motion information for the prediction step, a stereo camera was used to estimate the metric information of the detected features for measurement update step, and an OGM generated by a multilayer LIDAR was helpfully provided by Hata and Wolf (2016) to grant localization. As result, the proposed localization model has shown better accuracy and precision for estimation of orientation and lateral position but has presented difficulty on longitudinal position estimation. The reason for this may be related to road markings poorly mapped through LIDAR data and the lack of information provided to the localization algorithm by the proposed vision-based perception system. Also, it was noted that essentially every estimated pose was on the right side of the ground truth during all the test time, indicating a bias on lateral estimations due to a possible miscalibration of the stereo camera.

Although the achieved performance is still not enough for an autonomous navigation (absolute average euclidean error of $2.0042 \pm 1.2252\text{ m}$), it has shown that the proposed vision-based metric localization is consistent, and present enough precision for lane-level localization. Thus, it means that the proposed approach for road features detection and projection indeed can provide a low-level perception of the road environment.

7.1 Future Works

The main drawbacks of the proposed approach for road features detection on images are related to the complexity of a deep CNN architecture. First, a deep CNN implies a large VC-dimension, which makes the network prone to overfitting if not enough training data is provided. For example, Ferreira et al. (2018) showed by means of the SLT that more than 1.25 million samples of images would be necessary to have learning guarantee of the VGG16 network (SIMONYAN; ZISSERMAN, 2014), which is the same architecture used for the encoder part of the employed FCNN. This study enforces the idea that overfitting has very probably happened during the training process in this work. Second, running a deep CNN involves high computational cost, making it necessary to use a powerful GPU for real-time execution. In this sense, designing a simpler and still efficient architecture of FCNN must be considered to get closer to a solution of both aforementioned problems. Moreover, using a larger dataset for training and validation may provide better guarantee of generalization. For this, realistic simulated environments may be strong allies. For example, Dosovitskiy et al. (2017) introduced and open-sourced

CARLA simulator, which provides high-dimensional sensory observations, including color images and depth maps, not to mention segmented images in multiple classes. It could employ this simulator to get dataset for training, and test the FCNN performance on real scenarios to check the learning validity.

Another issue observed on experimental results was the difficulty of the trained FCNN to detect road features on images with strong shadows caused by trees. To deal with it, it must be considered using image representations robust against illumination effects, e.g. the hue and saturation channels of HSV color space.

Also, considering that the visual perception system must detect road features in a sequence of frames, time integration may be a powerful tool (as already reviewed in Section 3.1.1). For instance, there is the Recurrent Neural Network (RNN) concept that could be used to complement the employed FCNN. Moreover, still taking advantage of recurrent information, it could also be applied for localization. Although the proposed visual perception system is limited to the front scene observed by the vehicle, one could accumulate information from some previous perceptions and localization to merge consecutive projected point clouds to get more complete data from the vehicle's surroundings. Thus, it could improve the accuracy and precision of the localization estimations. Still, it would also be interesting to investigate dense disparity map estimation through CNN. In this way, it could be integrated into the same network used for road features detection, allowing the system to have the spatial perception of the road features by a single inference.

Regarding the projection of the detected road features to a point cloud, there are two main points that can be enhanced to get better results. First, improving the calibration of the stereo camera may solve the accuracy problem of the right curb reconstruction. Second, integrating IMU data during 3D projection may be considered to compensate projection tilts caused by trepidations of speed bumps. However, these improvements still would not solve the problem of the decreasing precision in function of the distance from the sensor. Nonetheless, since LIDAR can provide accurate and precise metric data, sensor fusion must not be discarded. Moreover, with LIDAR data, it would be possible to detect obstacles and regions of the environment that are not relevant for road features detection, providing valuable information for dynamic ROI definition for better efficiency.

Finally, after making possible improvements to the proposed road features detection and projection with aforestated suggestions, it would be interesting to generate a OGM with the same percetion system used for localization, in order to support better results for metric localization. Furthermore, it would also be interesting to investigate a topological localization model, where some road markings can be used as landmarks, and not be represented as obstacles in the map. In this way, the map representation would require much less memory space, which would be a problem for mapping a large area in a OGM with 0.1 *m* of resolution.

BIBLIOGRAPHY

- ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANE, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIEGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; ZHENG, X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 2016. mar 2016. Available from Internet: <<http://arxiv.org/abs/1603.04467>>.
- ÁLVAREZ, J. M.; LÓPEZ, A. M.; BALDRICH, R. Shadow Resistant Road Segmentation from a Mobile Monocular System. In: **Iberian Conference on Pattern Recognition and Image Analysis**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 9–16.
- AQEL, M. O. A.; MARHABAN, M. H.; Iqbal Saripan, M.; Bt Ismail, N. Review of visual odometry: types, approaches, challenges, and applications. **SpringerPlus**, 2016. v. 5, n. 1, p. 1897, dec 2016. ISSN 2193-1801.
- BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Scene Segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2017. p. 1–1, 2017. ISSN 0162-8828. Available from Internet: <<http://ieeexplore.ieee.org/document/7803544/>>.
- BAO, J.; GU, Y.; HSU, L.-T.; KAMIJO, S. Vehicle self-localization using 3D building map and stereo camera. In: **2016 IEEE Intelligent Vehicles Symposium (IV)**. Gothenburg: IEEE, 2016. p. 927–932. ISBN 978-1-5090-1821-5. Available from Internet: <<http://ieeexplore.ieee.org/document/7535499/>>.
- BAY, H.; TUYTELAARS, T.; Van Gool, L. SURF: Speeded Up Robust Features. In: . Springer, Berlin, Heidelberg, 2006. p. 404–417. Available from Internet: <http://link.springer.com/10.1007/11744023{}_32>.
- BROWN, D. C. Decentering Distortion of Lenses. **Annual Convention of the American Society of Photogrammetry**, 1965. D. Brown Associates, Inc., Eau Gallie, Florida, p. 444–462, 1965. Available from Internet: <https://www.asprs.org/wp-content/uploads/pers/1966journal/may/1966{}_may{}_444-462.pdf>.
- BRUBAKER, M. A.; GEIGER, A.; URTASUN, R. Map-Based Probabilistic Visual Self-Localization. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2016. v. 38, n. 4, p. 652–665, apr 2016. ISSN 0162-8828.
- CADENA, C.; CARLONE, L.; CARRILLO, H.; LATIF, Y.; SCARAMUZZA, D.; NEIRA, J.; REID, I.; LEONARD, J. J. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. **IEEE Transactions on Robotics**, 2016. v. 32, n. 6, p. 1309–1332, dec 2016. ISSN 1552-3098.
- CAI, Z.; FAN, Q.; FERIS, R. S.; VASCONCELOS, N. A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection. In: **European Conference on Computer Vision**. Amsterdam, Netherlands: Springer, Cham, 2016. p. 354–370.

CASELITZ, T.; STEDER, B.; RUHNKE, M.; BURGARD, W. Monocular camera localization in 3D LiDAR maps. In: **2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. : IEEE, 2016. p. 1926–1931. ISBN 978-1-5090-3762-9.

CHEN, Z.; CHEN, Z. RBNet: A Deep Neural Network for Unified Road and Road Boundary Detection. In: . Springer, Cham, 2017. p. 677–687. Available from Internet: <<http://link.springer.com/10.1007/978-3-319-70087-8{ }70>>.

CUI, D.; XUE, J.; ZHENG, N. Real-Time Global Localization of Robotic Cars in Lane Level via Lane Marking Detection and Shape Registration. **IEEE Transactions on Intelligent Transportation Systems**, 2016. v. 17, n. 4, p. 1039–1050, apr 2016. ISSN 15249050.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A large-scale hierarchical image database. In: **IEEE Conference on Computer Vision and Pattern Recognition**. Miami, USA: IEEE, 2009. p. 248–255. ISBN 978-1-4244-3992-8.

DOSOVITSKIY, A.; ROS, G.; CODEVILLA, F.; LOPEZ, A.; KOLTUN, V. CARLA: An Open Urban Driving Simulator. 2017. nov 2017. Available from Internet: <<http://arxiv.org/abs/1711.03938>>.

DU, X.; TAN, K. K. Comprehensive and Practical Vision System for Self-Driving Vehicle Lane-Level Localization. **IEEE Transactions on Image Processing**, 2016. v. 25, n. 5, p. 2075–2088, 2016. ISSN 10577149.

_____. Vision-based approach towards lane line detection and vehicle localization. **Machine Vision and Applications**, 2016. Springer Berlin Heidelberg, v. 27, n. 2, p. 175–191, feb 2016. ISSN 0932-8092.

DUMOULIN, V.; VISIN, F. A guide to convolution arithmetic for deep learning. 2016. mar 2016. Available from Internet: <<http://arxiv.org/abs/1603.07285>>.

ELFES, A. Using occupancy grids for mobile robot perception and navigation. **Computer**, 1989. v. 22, n. 6, p. 46–57, jun 1989. ISSN 0018-9162. Available from Internet: <<http://ieeexplore.ieee.org/document/30720/>>.

ENZWEILER, M.; GREINER, P.; KNOPPEL, C.; FRANKE, U. Towards multi-cue urban curb recognition. In: **IEEE Intelligent Vehicles Symposium (IV)**. Gold Coast City, Australia: IEEE, 2013. p. 902–907. ISBN 978-1-4673-2755-8.

FERREIRA, M. D.; CORRÊA, D. C.; NONATO, L. G.; MELLO, R. F. Designing architectures of convolutional neural networks to solve practical problems. **Expert Systems with Applications**, 2018. Elsevier, v. 94, p. 205–217, mar 2018. ISSN 0957-4174. Available from Internet: <<https://www.sciencedirect.com/science/article/pii/S0957417417307340{ }.WgGdhYNb25k.linkedinhttp://www.sciencedirect.com/science/article/pii/S0957417417307340>>.

FORSTER, C.; PIZZOLI, M.; SCARAMUZZA, D. SVO: Fast semi-direct monocular visual odometry. In: **2014 IEEE International Conference on Robotics and Automation (ICRA)**. : IEEE, 2014. p. 15–22. ISBN 978-1-4799-3685-4.

FOX, D. KLD-Sampling: Adaptive Particle Filters. In: **Advances in neural information processing systems**. 2002. p. 713–720. Available from Internet: <<http://papers.nips.cc/paper/1998-kld-sampling-adaptive-particle-filters.pdf>>.

FOX, D.; BURGARD, W.; THRUN, S. Markov Localization for Mobile Robots in Dynamic Environments. **Journal of Artificial Intelligence Research**, 1999. v. 11, p. 391–427, 1999.

FRITSCH, J.; KUHN, T.; GEIGER, A. A new performance measure and evaluation benchmark for road detection algorithms. In: **16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)**. IEEE, 2013. p. 1693–1700. ISBN 978-1-4799-2914-6. Available from Internet: <<http://ieeexplore.ieee.org/document/6728473/>>.

FUENTES-PACHECO, J.; RUIZ-ASCENCIO, J.; RENDÓN-MANCHA, J. M. Visual simultaneous localization and mapping: a survey. **Artificial Intelligence Review**, 2012. v. 43, n. 1, p. 55–81, 2012. ISSN 02692821.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. **Biological cybernetics**, 1980. v. 36, n. 4, p. 193–202, 1980.

GARCIA-FIDALGO, E.; ORTIZ, A. Vision-based topological mapping and localization methods: A survey. **Robotics and Autonomous Systems**, 2015. v. 64, p. 1–20, 2015. ISSN 09218890.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge: MIT Press, 2016. ISBN 9780262035613. Available from Internet: <<http://www.deeplearningbook.org/>>.

GRUYER, D.; BELAROUSSI, R.; REVILLOU, M. Map-aided localization with lateral perception. In: **2014 IEEE Intelligent Vehicles Symposium Proceedings**. : IEEE, 2014. p. 674–680. ISBN 978-1-4799-3638-0.

HARRIS, C.; STEPHENS, M. A combined corner and edge detector. In: **Alvey Vision Conference**. 1988. p. 147–151.

HATA, A. Y. **Road features detection and sparse map-based vehicle localization in urban environments**. 2016. Tese (Thesis) — University of São Paulo, 2016.

HATA, A. Y.; WOLF, D. F. Feature Detection for Vehicle Localization in Urban Environments Using a Multilayer LIDAR. **IEEE Transactions on Intelligent Transportation Systems**, 2016. v. 17, n. 2, p. 420–429, feb 2016.

HAYKIN, S. **Redes Neurais: Princípios e prática**. 2. ed. Porto Alegre: Bookman, 2001. 900 p. ISBN 978-85-7307-718-6.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep Residual Learning for Image Recognition. 2015. dec 2015. Available from Internet: <<http://arxiv.org/abs/1512.03385>>.

HEIKKILA, J.; SILVEN, O. A four-step camera calibration procedure with implicit image correction. In: **Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. IEEE Comput. Soc, 1997.

p. 1106–1112. ISBN 0-8186-7822-4. ISSN 1063-6919. Available from Internet: <<http://ieeexplore.ieee.org/document/609468/>>.

HERNANDEZ, J.; MARCOTEGUI, B. Filtering of Artifacts and Pavement Segmentation from Mobile LiDAR Data. In: **ISPRS Workshop Laserscanning**. Paris, France: , 2009. Available from Internet: <<https://hal-mines-paristech.archives-ouvertes.fr/hal-00833573>>.

HILLEL, A. B.; LERNER, R.; LEVI, D.; RAZ, G. Recent progress in road and lane detection: a survey. **Machine Vision and Applications**, 2014. Springer Berlin Heidelberg, v. 25, n. 3, p. 727–745, apr 2014. ISSN 0932-8092. Available from Internet: <<http://link.springer.com/10.1007/s00138-011-0404-2>>.

HUANG, A. S.; MOORE, D.; ANTONE, M.; OLSON, E.; TELLER, S. Finding multiple lanes in urban road networks with vision and lidar. **Autonomous Robots**, 2009. Springer US, v. 26, n. 2-3, p. 103–122, apr 2009. ISSN 0929-5593. Available from Internet: <<http://link.springer.com/10.1007/s10514-009-9113-3>>.

HUMENBERGER, M.; ZINNER, C.; WEBER, M.; KUBINGER, W.; VINCZE, M. A fast stereo matching algorithm suitable for embedded real-time systems. **Computer Vision and Image Understanding**, 2010. Academic Press, v. 114, n. 11, p. 1180–1202, nov 2010. ISSN 1077-3142. Available from Internet: <<http://www.sciencedirect.com/science/article/pii/S1077314210000895>>.

J.G. Fryer; D.C. Brown. Lens Distortion for Close-Range Photogrammetry. **Photogrammetric engineering and remote sensing**, 1986. v. 52(1), n. 1, p. 51–58, 1986.

JO, K.; JO, Y.; SUHR, J. K.; JUNG, H. G.; SUNWOO, M. Precise Localization of an Autonomous Car Based on Probabilistic Noise Models of Road Surface Marker Features Using Multiple Cameras. **IEEE Transactions on Intelligent Transportation Systems**, 2015. v. 16, n. 6, p. 3377–3392, dec 2015. ISSN 1524-9050.

KATRAMADOS, I.; CRUMPLER, S.; BRECKON, T. P. Real-Time Traversable Surface Detection by Colour Space Fusion and Temporal Analysis. In: **International Conference on Computer Vision Systems**. Liège, Belgium: Springer, Berlin, Heidelberg, 2009. p. 265–274.

KINGMA, D. P.; BA, J. Adam: A Method for Stochastic Optimization. 2014. dec 2014. Available from Internet: <<http://arxiv.org/abs/1412.6980>>.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. **Advances in Neural Information Processing Systems**, 2012. p. 1097–1105, 2012.

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation Applied to Handwritten Zip Code Recognition. **MIT Press Journals**, 1989. v. 1, p. 541–551, 1989.

LEVINSON, J.; ASKELAND, J.; BECKER, J.; DOLSON, J.; HELD, D.; KAMMEL, S.; KOLTER, J. Z.; LANGER, D.; PINK, O.; PRATT, V.; SOKOLSKY, M.; STANEK, G.; STAVENS, D.; TEICHMAN, A.; WERLING, M.; THRUN, S. Towards fully autonomous driving: Systems and algorithms. In: **2011 IEEE Intelligent Vehicles Symposium (IV)**. : IEEE, 2011. p. 163–168. ISBN 978-1-4577-0890-9.

- LEVINSON, J.; MONTEMERLO, M.; THRUN, S. Map-Based Precision Vehicle Localization in Urban Environments. **Robotics: Science and Systems III**, 2007. Robotics: Science and Systems Foundation, p. 121–128, jun 2007. ISSN 2330765X. Available from Internet: <<http://www.roboticsproceedings.org/rss03/p16.pdf>>.
- LI, H.; ZHAO, R.; WANG, X. Highly Efficient Forward and Backward Propagation of Convolutional Neural Networks for Pixelwise Classification. 2014. dec 2014. Available from Internet: <<http://arxiv.org/abs/1412.4526>>.
- LIU, L.; WU, T.; FANG, Y.; HU, T.; SONG, J. A smart map representation for autonomous vehicle navigation. In: **2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)**. : IEEE, 2015. p. 2308–2313. ISBN 978-1-4673-7682-2.
- LIU, X.; DAI, B.; SONG, J.; HE, H.; ZHANG, B. Real-Time Long-Range Lane Detection and Tracking for Intelligent Vehicle. In: **2011 Sixth International Conference on Image and Graphics**. IEEE, 2011. p. 654–659. ISBN 978-1-4577-1560-0. Available from Internet: <<http://ieeexplore.ieee.org/document/6005947/>>.
- LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. IEEE, 2015. p. 3431–3440. ISBN 978-1-4673-6964-0. Available from Internet: <<http://ieeexplore.ieee.org/document/7298965/>>.
- LOWE, D. Object recognition from local scale-invariant features. In: **Proceedings of the Seventh IEEE International Conference on Computer Vision**. : IEEE, 1999. v. 2, p. 1150–1157. ISBN 0-7695-0164-8.
- LU, W.; Rodriguez F., S. A.; SEIGNEZ, E.; REYNAUD, R. Monocular multi-kernel based lane marking detection. In: **The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent**. IEEE, 2014. p. 123–128. ISBN 978-1-4799-3669-4. Available from Internet: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6917447>>.
- LUXBURG, U. von; SCHOELKOPF, B. Statistical Learning Theory: Models, Concepts, and Results. 2008. oct 2008. Available from Internet: <<http://arxiv.org/abs/0810.4752>>.
- MENDES, C. C. T.; FREMONT, V.; WOLF, D. F. Exploiting fully convolutional neural networks for fast road detection. **2016 IEEE International Conference on Robotics and Automation (ICRA)**, 2016. n. May, p. 3174–3179, 2016. ISSN 10504729.
- MORAVEC, H. P. Sensor Fusion in Certainty Grids for Mobile Robots. **AI Magazine**, 1988. v. 9, n. 2, p. 61, jun 1988. ISSN 0738-4602. Available from Internet: <<https://www.aaai.org/ojs/index.php/aimagazine/article/view/676>>.
- NEWCOMBE, R. A.; LOVEGROVE, S. J.; DAVISON, A. J. DTAM: Dense tracking and mapping in real-time. In: **2011 International Conference on Computer Vision**. IEEE, 2011. p. 2320–2327. ISBN 978-1-4577-1102-2. Available from Internet: <<http://ieeexplore.ieee.org/document/6126513/>>.
- NISTER, D.; NARODITSKY, O.; BERGEN, J. Visual odometry. In: **Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and**

Pattern Recognition, 2004. CVPR 2004. IEEE, 2004. v. 1, p. 652–659. ISBN 0-7695-2158-4. Available from Internet: <<http://ieeexplore.ieee.org/document/1315094/>>.

ONIGA, F.; NEDEVSKI, S.; MEINECKE, M. M.; TO, T. B. Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo. In: **IEEE Intelligent Transportation Systems Conference**. Bellevue, WA, USA: IEEE, 2007. p. 859–865. ISBN 978-1-4244-1395-9. Available from Internet: <<http://ieeexplore.ieee.org/document/4357734/>>.

POMERLEAU, F.; COLAS, F.; SIEGWART, R. A Review of Point Cloud Registration Algorithms for Mobile Robotics. **Foundations and Trends in Robotics**, 2015. Now Publishers, Inc., v. 4, n. 1, p. 1–104, 2015. Available from Internet: <<http://www.nowpublishers.com/article/Details/ROB-035>>.

PRADEEP, V.; MEDIONI, G.; WEILAND, J. Piecewise Planar Modeling for Step Detection using Stereo Vision. In: **Workshop on Computer Vision Applications for the Visually Impaired**. Marseille, France: , 2008. Available from Internet: <<https://hal.inria.fr/inria-00325448/>>.

ROSS, P. E. **Velodyne Unveils Monster Lidar With 128 Laser Beams**. 2017. Available from Internet: <<https://spectrum.ieee.org/cars-that-think/transportation/sensors/velodyne-unveils-monster-lidar-with-128-laser-beams>>.

ROSTEN, E.; DRUMMOND, T. Machine Learning for High-Speed Corner Detection. In: . Springer, Berlin, Heidelberg, 2006. p. 430–443. Available from Internet: <http://link.springer.com/10.1007/11744023_{ }34>.

SCHARSTEIN, D.; SZELISKI, R. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. **International Journal of Computer Vision**, 2002. Kluwer Academic Publishers, v. 47, n. 1/3, p. 7–42, 2002. ISSN 09205691. Available from Internet: <<http://link.springer.com/10.1023/A:1014573219977>>.

SCHMIDHUBER, J. Deep Learning in Neural Networks: An overview. **Neural Networks**, 2014. v. 61, p. 85–117, 2014. ISSN 18792782.

SCHREIBER, M.; KNÖPPEL, C.; FRANKE, U. LaneLoc: Lane marking based localization using highly accurate maps. **IEEE Intelligent Vehicles Symposium, Proceedings**, 2013. n. Iv, p. 449–454, 2013.

SEBSADJI, Y.; TAREL, J.-P.; FOUCHER, P.; CHARBONNIER, P. Robust road marking extraction in urban environments using stereo images. In: **2010 IEEE Intelligent Vehicles Symposium**. IEEE, 2010. p. 394–400. ISBN 978-1-4244-7866-8. Available from Internet: <<http://ieeexplore.ieee.org/document/5548075/>>.

SEGAL, A. V.; HAEHNEL, D.; THRUN, S. Generalized-ICP. In: **Robotics: Science and Systems**. Seattle: , 2009. Available from Internet: <http://www.robots.ox.ac.uk/{~}avsegal/resources/papers/Generalized_{ }ICP.pdf>.

SHINZATO, P. Y.; GOMES, D.; WOLF, D. F. Road estimation with sparse 3D points from stereo data. In: **17th International IEEE Conference on Intelligent Transportation Systems (ITSC)**. Qingdao, China: IEEE, 2014. p. 1688–1693. ISBN 978-1-4799-6078-1. Available from Internet: <<http://ieeexplore.ieee.org/document/6957936/>>.

SHINZATO, P. Y.; GRASSI, V.; OSORIO, F. S.; WOLF, D. F. Fast visual road recognition and horizon detection using multiple artificial neural networks. In: **IEEE Intelligent Vehicles Symposium**. Alcal de Henares, Spain: IEEE, 2012. p. 1090–1095. ISBN 978-1-4673-2118-1. Available from Internet: <<http://ieeexplore.ieee.org/document/6232175/>>.

SHINZATO, P. Y.; SANTOS, T. C. dos; ROSERO, L. A.; RIDEL, D. A.; MASSERA, C. M.; ALENCAR, F.; BATISTA, M. P.; HATA, A. Y.; OSORIO, F. S.; WOLF, D. F. CaRINA dataset: An emerging-country urban scenario benchmark for road detection systems. In: **2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)**. IEEE, 2016. p. 41–46. ISBN 978-1-5090-1889-5. Available from Internet: <<http://ieeexplore.ieee.org/document/7795529/>>.

SILVA, I. N. da; SPATTI, D. H.; FLAUZINO, R. A. **Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas: Curso Prático**. São Paulo: Artliber, 2010. ISBN 978-85-88098-53-4.

SIMONITE, T. **Self-Driving Cars’ Spinning-Laser Problem**. 2017. Available from Internet: <<https://www.technologyreview.com/s/603885/autonomous-cars-lidar-sensors/?set=603886>>.

SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014. sep 2014. Available from Internet: <<http://arxiv.org/abs/1409.1556>>.

SZELISKI, R. **Computer Vision**. London: Springer London, 2011. (Texts in Computer Science). ISSN 10636919. ISBN 978-1-84882-934-3. Available from Internet: <http://research.microsoft.com/en-us/um/people/szeliski/book/drafts/szelski_{__}20080330am_{__}draft.pdf<http://link.springer.com/10.1007/978-1-84882-935-0>>.

TEICHMANN, M.; WEBER, M.; ZOELLNER, M.; CIPOLLA, R.; URTASUN, R. MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. 2016. dec 2016. Available from Internet: <<http://arxiv.org/abs/1612.07695>>.

THRUN, S. Robotic Mapping: A Survey. **Science**, 2002. v. 298, n. February, p. 1–35, 2002. ISSN 00368075.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic Robotics**. Cambridge: The MIT Press, 2006. 672 p. ISBN 978-0-262-20162-9.

THRUN, S.; FOX, D.; BURGARD, W.; DELLAERT, F. Robust Monte Carlo localization for mobile robots. **Artificial Intelligence**, 2001. v. 128, n. 1, p. 99–141, 2001. ISSN 00043702.

TRUCCO, E.; VERRI, A. **Introductory techniques for 3-D computer vision**. : Prentice Hall, 1998. ISBN 0132611082.

VEIT, T.; TAREL, J.-P.; NICOLLE, P.; CHARBONNIER, P. Evaluation of Road Marking Feature Extraction. In: **11th International IEEE Conference on Intelligent Transportation Systems**. Beijing, China: IEEE, 2008. p. 174–181. ISBN 978-1-4244-2111-4.

WEBB, A. **Google and Uber Are Fighting Over Lidar Technology. What Is It?** 2017. Available from Internet: <<https://www.bloomberg.com/news/articles/2017-02-24/google-and-uber-are-fighting-over-lidar-technology-what-is-it>>.

WEI, L.; CAPPELLE, C.; RUICHEK, Y. Horizontal/vertical LRFs and GIS maps aided vehicle localization in urban environment. In: **16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)**. IEEE, 2013. p. 809–814. ISBN 978-1-4799-2914-6. Available from Internet: <<http://ieeexplore.ieee.org/document/6728331/>>.

WOLCOTT, R. W.; EUSTICE, R. M. Visual localization within LIDAR maps for automated urban driving. In: **2014 IEEE/RSJ International Conference on Intelligent Robots and Systems**. : IEEE, 2014. p. 176–183. ISBN 978-1-4799-6934-0.

_____. Fast LIDAR localization using multiresolution Gaussian mixture maps. In: **2015 IEEE International Conference on Robotics and Automation (ICRA)**. : IEEE, 2015. p. 2814–2821. ISBN 978-1-4799-6923-4.

XIAO, L.; LIU, L.; WU, T.; DAI, B. Monocular Localization Within Manually Annotated LIDAR Maps. In: **2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)**. : IEEE, 2016. p. 252–256. ISBN 978-1-5090-0768-4.

ZEILER, M. D.; KRISHNAN, D.; TAYLOR, G. W.; FERGUS, R. Deconvolutional networks. In: **2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. IEEE, 2010. p. 2528–2535. ISBN 978-1-4244-6984-0. Available from Internet: <<http://ieeexplore.ieee.org/document/5539957/>>.

ZHANG, G.; ZHENG, N.; CUI, C.; Yuzhen Yan; Zejian Yuan. An efficient road detection method in noisy urban environment. In: **Intelligent Vehicles Symposium**. Xian, China: IEEE, 2009. p. 556–561. ISBN 978-1-4244-3503-6.

ZHANG, Z. A flexible new technique for camera calibration. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2000. v. 22, n. 11, p. 1330–1334, 2000. ISSN 01628828. Available from Internet: <<http://ieeexplore.ieee.org/document/888718/>>.

ZIEGLER, J.; LATEGAHN, H.; SCHREIBER, M.; KELLER, C. G.; KNOPPEL, C.; HIPPE, J.; HAUEIS, M.; STILLER, C. Video Based Localization for BERTHA. **2014 IEEE Intelligent Vehicles Symposium Proceedings**, 2014. n. Iv, p. 1231–1238, 2014.

Appendix

APPENDIX A – PUBLICATION

A.1 Related Paper

HORITA, L. R. T.; GRASSI Jr., V. Employing a Fully Convolutional Neural Network for Road Marking Detection. In: **IEEE 2017 14th Latin American Robotics Symposium and 2017 5th Brazilian Symposium on Robotics (LARS-SBR)**. Curitiba: 2017. p. 1-6.

A.2 Others

NAKAMURA, A. T. M.; HORITA, L. R. T.; GRASSI Jr., V. Aplicabilidade dos Métodos Visuais de Detecção de Pedestres para Veículos Inteligentes. In: **XIII Simpósio Brasileiro de Automação Inteligente (SBAI)**. Porto Alegre: 2017. p. 1616-1622.

NAKAMURA, A. T. M.; HORITA, L. R. T.; GRASSI Jr., V. A Stereo Cameras Setup for Pedestrian Detection Enhancement. In: **IEEE 5th International Symposium on Robotics and Intelligent Sensors (IEEE-IRIS2017)**. Ottawa, Canada: 2017.