



HAL
open science

3D urban scene understanding by analysis of LiDAR, color and hyperspectral data

David Duque-Arias

► **To cite this version:**

David Duque-Arias. 3D urban scene understanding by analysis of LiDAR, color and hyperspectral data. Signal and Image Processing. Université Paris sciences et lettres, 2021. English. NNT : 2021UPSLM028 . tel-03434199

HAL Id: tel-03434199

<https://pastel.hal.science/tel-03434199>

Submitted on 18 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à MINES ParisTech

**3D urban scene understanding by analysis of LiDAR, color
and hyperspectral data**

Compréhension de scènes urbaines 3D par analyse de données LiDAR, colorées et
hyperspectrales

Soutenue par

David Duque-Arias

Le 28 Octobre 2021

École doctorale n°621

**Ingénierie des Systèmes,
Matériaux, Mécanique,
Énergétique - ISMME**

Spécialité

Morphologie mathématique

Composition du jury :

Pascal MONASSE

Professeur, Ecole Nationale des Ponts et Chaussées *Président*

Sébastien LEFEVRE

Professeur, Université de Bretagne-Sud *Rapporteur*

Andreas NÜCHTER

Professeur, University of Würzburg (Allemagne) *Rapporteur*

Paul CHECCHIN

Professeur, Université Clermont Auvergne *Examineur*

Santiago VELASCO-FORERO

Chargé de recherche, MINES ParisTech *Examineur*

Beatriz MARCOTEGUI

Professeur, MINES ParisTech *Directeur de thèse*

Jean-Emmanuel DESCHAUD

Chargé de recherche, MINES ParisTech *Co-directeur*



Ph.D. Thesis

3D urban scene understanding by analysis of LiDAR, color and hyperspectral data

*Compréhension de scènes urbaines 3D par analyse de données
LiDAR, colorées et hyperspectrales*

Ph.D. candidate: David Duque-Arias

Advisors: Ph.D. Santiago Velasco-Forero
Ph.D. Jean-Emmanuel Deschaud
Prof. Beatriz Marcotegui

Thesis defense: 2021-10-28

Acknowledgements

First of all, I would like to express my gratitude to my thesis advisors: Beatriz Marcotegui, Santiago Velasco-Forero and Jean-Emmanuel Deschaud for accepting me as their student. I really appreciate the opportunity of working with them and learning from their experience and expertise. Many useful discussions and research questions guided me during my PhD study. Also many thanks to my committee members, François Goulette, Sébastien Lefevre, Andreas Nuchter, Pascal Monasse and Paul Checchin for helping me to improve the quality of the manuscript.

I would also like to express my gratitude to the REPLICA Project FUI 24 for providing the funding of this thesis.

I am so grateful of knowing amazing and friendly people in the Center for Mathematical Morphology (CMM) during there these years. I would like to thank to: Etienne, Samy, Michel, Petr, Bruno, François, Jesus and Jose. A very special thanks to Anne-Marie for being so friendly, helpful and for teaching me new French expressions. I also want to thank to all the students in the CMM: Leo, Valentin, Tarek, Martin, Mateus, Thomas, Joao, Ayoub and Kaiwen.

Living in Fontainebleau (Avon) has been an incredible and enriching experience. Thanks to the *Hippies* group for providing a friendly environment that helped to feel less the missing from my country. I would like to thank to Andres and Anais who opened both their home and heart to me when I first arrived to France. Thanks to Eric and Gabi for their friendship and support, especially in the *dernière ligne droite* of the manuscript.

Very special thanks to my parents and my brother for being always the inspiration and the unconditional support.

And last but not least, I would like to thank to my love Dino, that supported me in the choice of coming to France to study a PhD. Much of this achievement comes from her help and encouragement when things became hard.

Abstract

Point clouds have attracted the interest of the research community over the last years. Initially, they were mostly used for remote sensing applications. More recently, thanks to the development of low-cost sensors and the publication of some open source libraries, they have become very popular and have been applied to a wider range of applications. One of them is the autonomous vehicle where many efforts have been made in the last century to make it real. A very important bottleneck nowadays for the autonomous vehicle is the evaluation of the proposed algorithms. Due to the huge number of possible scenarios, it is not feasible to perform it in real life. An alternative is to simulate virtual environments where all possible configurations can be set up beforehand. However, they are not as realistic as the real world is. In this thesis, we studied the pertinence of including hyperspectral images in the creation of new virtual environments. Furthermore, we proposed new methods to improve 3D scene understanding for autonomous vehicles. During this research, we addressed the following topics.

Firstly, we analyzed the spectrum in color and hyperspectral images because it provides a description about the electromagnetic radiation at different frequencies. Some applications rely only on visible colors. In other cases, such as the characterization of materials, the study of the invisible range is required. For this purpose, we proposed a simplified spectrum representation that preserves its diversity, the Graph-based color lines (GCL) model.

Secondly, we studied the integration of hyperspectral images, color images and point clouds in urban scenes. The analysis was carried out by using the data acquired during this thesis in the context of the REPLICA project FUI 24. We inspected spectral signatures of different objects and reflectance histograms of the images. The obtained results demonstrate that urban scenes are challenging scenarios for current technology of hyperspectral cameras due to the presence of uncontrolled light conditions and moving actors.

Thirdly, we worked with 3D point clouds from urban scenes that have proved to be a reliable type of data, much less sensitive to illumination variations than cameras. They are more accurate than color images and permit to obtain precise 3D models of urban environments. Deep learning techniques are very popular in this domain. A key element of these techniques is the loss function that drives the optimization process. We proposed two new loss functions to perform semantic segmentation tasks: power Jaccard loss and hierarchical loss. They obtained a higher performance in evaluated scenarios than classical losses not only in 3D point clouds but also in color and gray scale images. Moreover, we proposed a new dataset (Paris Carla 3D Dataset) composed of synthetic and real point clouds from urban scenes. It is expected to be used by the research community for different automatic tasks such as semantic segmentation, instance segmentation and scene completion.

Finally, we conducted a detailed analysis of the influence of RGB features in semantic segmentation of urban point clouds. We compared several training scenarios and identified that color systematically improves the performance in certain classes. It demonstrates that including a more detailed description of the spectrum, when the hyperspectral cameras technology increases its sensitivity, can be useful to improve scene description of urban scenes.

Keywords: Point clouds, Deep Learning, loss functions, image representation, semantic segmentation, hyperspectral images, Image processing.

Contents

List of Figures	ix
List of Tables	xv
1 Introduction	1
1.1 Résumé	1
1.2 Motivation	1
1.3 REPLICA project	2
1.4 Goals of this thesis	3
1.5 Thesis outline	5
1.6 Associated publications	5
2 Data used in this thesis	7
2.1 Résumé	7
2.2 Introduction	7
2.3 Images	8
2.3.1 Datasets	8
2.3.2 Multispectral/Hyperspectral Images	12
2.4 Point clouds	16
2.4.1 Definition	16
2.4.2 Datasets	16
2.4.3 2D projections	19
2.4.4 Neighborhood	23
2.5 REPLICA data	26
2.5.1 System description	27
2.5.2 Acquisitions	30
2.6 Paris-Carla 3D dataset	32
2.6.1 CARLA data	32
2.6.2 Paris data	32
2.6.3 Dataset properties	32
2.7 Conclusions	34

3	Graph based color lines (GCL)	41
3.1	Résumé	41
3.2	Introduction	41
3.3	State of the art	43
3.3.1	Color representation	43
3.3.2	Metrics used in this chapter	47
3.4	Representation of color images	50
3.4.1	GCL definition	50
3.4.2	Experimental Results	53
3.4.3	Including spatial information to GCL	58
3.4.4	Discussion about color images	62
3.5	Extension to MSI and HSI	63
3.5.1	GCL on HSI/MSI	64
3.5.2	Experimental results	66
3.5.3	Improvements to GCL in HSI/MSI	67
3.5.4	Discussion about MSI / HSI	70
3.6	Conclusions	71
4	Study of REPLICA data	73
4.1	Résumé	73
4.2	Introduction	73
4.3	Analysis of hyperspectral data	73
4.3.1	Spectrum and reflectance analysis	74
4.3.2	Integration with point clouds	94
4.4	Most relevant findings	96
4.5	Conclusions	97
5	Power Jaccard losses for semantic segmentation	99
5.1	Résumé	99
5.2	Introduction	99
5.3	Overview of DL	99
5.4	Loss functions	102
5.5	Power Jaccard loss	103
5.5.1	Jaccard index	103
5.5.2	Our proposal: Power Jaccard Loss	104
5.5.3	Derivatives of Power Jaccard	105
5.6	Experimental design	106
5.6.1	Grayscale images	106
5.6.2	Color images	107
5.6.3	Point clouds	107
5.7	Results	108
5.7.1	Grayscale images	108
5.7.2	RGB images	110
5.7.3	Point clouds projections	112
5.8	Discussion	113
5.9	Conclusions	115

6	Segmentation of point clouds	117
6.1	Résumé	117
6.2	Introduction	117
6.3	Semantic segmentation	117
6.3.1	Related Work	118
6.3.2	Proposed approaches	123
6.3.3	Transfer learning	135
6.4	Hierarchical learning for semantic segmentation	135
6.4.1	Related work	136
6.4.2	Learning based on a class hierarchy	137
6.5	Instance segmentation	141
6.5.1	Related Work	141
6.5.2	Baseline for PC3D dataset	142
6.6	Color in urban point clouds	144
6.6.1	Experiments	144
6.6.2	Analysis of results	145
6.7	Discussion	148
6.8	Conclusions	149
7	Conclusions and perspectives	151
7.1	Résumé	151
7.2	Conclusions	151
7.3	Perspectives	153
	Bibliography	155

List of Figures

1.1	REPLICA 24 FUI project brings together six industrial partners: OPTIS, Terra3D, Renault, PSA group, ALL4TEC, OKTAL and one academic partner: ARMINES/MINES ParisTech.	3
1.2	Diagram with the main structure of this thesis. The contributions are presented in the boxes with underlined text: Graph-based color lines and analysis of hyperspectral data (Chapter 3); power Jaccard loss (Chapter 5); hierarchical losses and study of RGB influence in semantic segmentation of point clouds (Chapter 6).	4
2.1	Example of color image and its corresponding ground truth from DAVIS dataset [1]	9
2.2	Example of two images from Barcelona dataset [2]	9
2.3	Example of color image and its corresponding ground truth from BSD-500 dataset [3]	10
2.4	Images from generated MNIST dataset for segmentation	11
2.5	Selected classes of example image from Cityscapes dataset [4]	11
2.6	Example image from CVPPP dataset [5] and its ground truth to perform leaves counting and binary segmentation.	12
2.7	Example of aerial images from Toronto University dataset [6](color and ground truth). The dataset is composed of three sub-datasets in order to perform different binary segmentation tasks. We selected two of them to perform some experiments in our work: the Massachusetts roads dataset and the Massachusetts building dataset.	13
2.8	Representation of a hyperspectral image	14
2.9	Images from University of the Negev dataset [7] with RGB colors computed from hyperspectral data.	15
2.10	Images from powders dataset [8].	15
2.11	Example of 3D point cloud with color and intensity information from rue Soufflot, in Paris.	17
2.12	Example of a 3D point cloud from Semantic KITTI dataset [9]. Color scale of most visible classes: Road is purple, traffic signs are blue, cars are orange, vegetation is green, low vegetation is light green and buildings are blue.	18
2.13	Example of 3D point cloud from Street Scenes dataset [10]. Color scale: buildings are yellow, cars are red, ground is gray, vegetation is green, poles are orange and unlabeled points are white.	19
2.14	Projections of point cloud from Semantic KITTI dataset coloured with ground truth labels.	21

2.15	Simplified representation of Velodyne with $n = 4$. FOV_{top} and FOV_{down} are the maximum and minimum angles of the field of view. According to the number of layers and angle values, dimension of the rectangle changes. In most common commercial scanners, usually $n = 32, 64, 128$ and FOV angles are between 15 and 30 degrees. Recently, new scanners with $n = 8, 16$ are available	22
2.16	Voxelization of a car varying voxel size. Color scale corresponds to x coordinate of voxels center.	24
2.17	Neighborhood of two points (in red) from an outdoor scene point cloud.	25
2.18	L3D2 platform equipped with several sensors: LiDAR sensor, location system and a set of color and multispectral cameras. Each sensor has a reference frame with respect to is acquired the data.	28
2.19	Sensors mounted on the roof of L3D2 vehicle: 1) Velodyne HDL-32E LiDAR; 2) Ladybug camera; 3) Antenna of GPS; 4) Hyperspectral cameras HS02 and HS03. . .	29
2.20	Raw color images acquired with LadyBug camera at Rue Soufflot in Paris. Each camera acquires a part of the 360°sphere (about 90 %, according to manufacturer). .	36
2.21	Raw image acquired with HS02 camera from Ecole des Mines. A branch of the tree on the top of the image was zoomed in order to visualize the mosaic pattern present in both HS02 and HS03 cameras.	37
2.22	Area where acquisitions were performed, close to Luxembourg gardens and Pantheon, in the 5th arrondissement of Paris. The red line indicates the streets followed by the L3D2 vehicle.	37
2.23	Dense point cloud from the first acquisition. It contains 10 million points and includes intensity (Fig. 2.23(a)) and color information (Fig. 2.23(b)). Other fields are also included in the file, such as: scan angle, GPS time, sensor position, vertical laser angle, laser index and frame index.	38
2.24	Images in false RGB colors from HS03 camera. Three different exposure times were tested in the fourth acquisition: 5 ms, 50 ms and 100 ms.	39
3.1	Comparison between DCL and GCL. Our model fits spectral representation by piecewise lines.	42
3.2	RGB color space	43
3.3	Timeline with the number of works on color between 1984 and 2018.	45
3.4	Visual comparison of superpixels. Left column: watershed superpixel [11]; center column: Turbopixels [12]; right column: SLIC [13]. Image taken from [11]	47
3.5	Steps for computing simplified spectral representation in GCL model.	50
3.6	Gaussian means of spectral representation of cropped section in Figure 3.1 (a). . . .	51
3.7	First two DCLs of cropped section Figure 3.1 (a).	52
3.8	Illustrative example of Algorithm 2 to compute top-3 color lines.	53
3.9	Top-2 GCL of cropped section from Figure 3.1 (a), computed in the RGB cube and projected into 2D planes for visualization purposes.	54
3.10	Illustrative example of color points projection on a GCL	55
3.11	PSNR varying the number of GCL and hemispheres from Table 3.1.	56
3.12	PSNR and MSE in two videos from [1]. PSNR (blue) and MSE (red).	57
3.13	Images from boxing-fisheye [1].	57
3.14	Images from mallard-fly [1].	58

3.15	Modified diagram including spatial information to the simplified representation presented in Figure 3.5.	59
3.16	Dimensionality reduction with PCA.	59
3.17	h-maxima transformation [14].	60
3.18	Example of image representation varying the weight of spatial information s . It is observed how the shape of regions changes according to s value. The use of $s > 1$ leads to regions grouping pixels based on their spatial proximity and less preservation of contours.	61
3.19	Simplified representation using a hyperspectral image from powders dataset with 79 Gaussians. PSNR: 32.94 dB.	64
3.20	PCA and projection of 3D-histogram of hyperspectral image	65
3.21	Clustering of color points using the third approach introduced in Section 3.5.1.	66
3.22	Simplified representation of hyperspectral image	68
3.23	Example of pixel clustering in Gaussians	69
3.24	Comparison of simplified representations	69
4.1	Traffic signs from the first acquisition of rue Soufflot, Paris. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from HS03 camera, in blue, and spectral signatures from Ladybug camera, in red. Third column: Spectral signatures from HS02 camera, in blue, and spectral signatures of known materials [15, 16], in other colors.	75
4.2	Spectral signatures of other objects from the first acquisition. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from HS03 camera, in blue, and spectral signatures from Ladybug camera, in red. Third column: Spectral signatures from HS02 camera, in blue, and spectral signatures of known materials [15, 16], in other colors.	77
4.3	Histogram of HS images from traffic-sign presented in Figure 4.1(f). RGB image was computed from image acquired in the visible range with HS03 camera.	79
4.4	Distribution of reflectance values in HS images in first acquisition. Minimum (red), mean (blue), and maximum (orange) values are presented. Standard deviation is displayed with a vertical blue line. It is seen how the mean, minimum and maximum values of all channels are very similar in all channels of both cameras.	80
4.5	Comparison of spectral signatures of traffic sign from first (blue) and second (green) acquisitions. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from visible range camera and spectral signatures from Ladybug camera in red. Third column: Spectral signatures from NIR range.	82
4.6	Histograms of first (blue) and second (green) acquisitions. Each line is a channel histogram of all acquired images. In both acquisitions, histograms have similar values in all channels. In the first acquisition, an over-saturation effect is present around 0.5, corrected for the second acquisition. However, spectral data is still concentrated around 0.1 in both ranges.	83
4.7	Expanded histograms based on threshold criterion. Green lines are original histograms from second acquisition without modifications. In both ranges, histograms were thresholded using $t = [0.2, 0.25, 0.4, 0.6]$	86

4.8	Expanded histograms based on percent criterion. Green lines are original histograms from second acquisition without modifications. In both ranges, histograms were normalized using $p = [0.98, 0.95, 0.9, 0.85]$	86
4.9	Spectral signatures using the proposed normalization methods with data from second acquisition. Autumn 2019 v1 in blue, Autumn 2019 v2 in green, normalization by threshold using $t = 0.2$ in cyan and normalization by percentage using $p = 0.95$ in orange.	88
4.10	Comparison of spectral signatures of same objects in first three acquisitions. First acquisition in blue, second acquisition in green and third acquisition in green.	90
4.11	Histograms of first (blue), second (green) and third (magenta) acquisitions. Each line is a channel histogram of all acquired images. In the first acquisition, an over saturation effect is present around 0.5. The second and third acquisitions have same post-processing stage. One may observe that even if the last acquisition was conducted in summer, reflectance values are still low in all channels. In NIR range, they are even lower than in previous acquisitions.	91
4.12	Spectral signatures of same objects varying exposure time between 5ms (blue), 50ms (red) and 100ms (green) of fourth acquisition.	93
4.13	Histograms of images acquired in fourth acquisition with different exposure times: 100ms (blue), 50ms (red) and 5ms (green).	94
4.14	Crop of selected point cloud to study the offset between 2D and 3D data. The points seen in the same HS image are painted in brown and three of the manually segmented traffic-signs in green, orange and red.	95
4.15	Projection of 3D points into the closest HS image based on timestamp information. Images a), b) and c) are not the same.	96
5.1	Simplified diagram of training stage of Convolutional Neural Network (CNN) in a semantic segmentation application.	101
5.2	Comparison of some classical loss functions (dotted lines) and our proposed Power Jaccard loss (solid line). Binary Cross-Entropy. For Focal BCE with $\gamma = 2$. Vertical red line indicates the ground truth $y = 1$. Our proposal reduces the relative loss for well-classified examples.	102
5.3	Incidence of parameter p in power Jaccard loss. Vertical red line indicates the ground truth value of $y = 1$	104
5.4	Variation of minimum of power Jaccard loss according to p value.	106
5.5	Point cloud with ground truth from Street 3D [10] dataset and the corresponding BEV projections.	109
5.6	Multiclass segmentation using MNIST dataset. From left to right: 1) Original gray scale image; 2) Ground truth generated from gray scale image; 3) Prediction using best obtained model; 4) Errors between prediction and ground truth.	111
5.7	Example of Prediction on Figure 2.5 in Cityscapes. Road (blue), person (red), car (green) and background (black). Note that Power Jaccard performs better on smaller classes than the classical one. Quantitative results are given in Table 5.4.	113
6.1	Number of publications by year in Google Scholar with the query “urban 3d point cloud segmentation”	118
6.2	PointNet architecture proposed by [17]	120

6.3	PointNet++ architecture proposed by [18]	121
6.4	Comparison 2D convolutions and KP Convolutions. Image taken from [19].	123
6.5	Rate of points projected in a pixel where other pixel has already been projected. Experiments performed with first 300 point clouds from validation set from Semantic KITTI dataset.	124
6.6	Crops of 64x200 extracted from 64x1024 spherical projections from semantic KITTI dataset.	125
6.7	Accuracy varying back-projection method of spherical projections to 3D. Experiments were carried with first 300 point clouds from validation set of Semantic KITTI as follows: <i>all points</i> : Same label for all points projected in the same pixel; k_0 : Label only point closest point (other points remain unlabeled); k_n : Perform KNN with $K = n$ to label non projected points.	127
6.8	Workflow of proposed Spherical DZNet method during SHREC 2020 challenge . . .	128
6.9	Crops of spherical projections of point cloud from Street 3D dataset.	129
6.10	Proposed Network architecture. Yellow boxes represent Conv2D layers of 5x5 with strides (1,2), red boxes Conv2D of 1x1 followed by a dropout of 0.1. Blue color means Upsampling2D layer. Violet arrows are used to illustrate the concatenation operator. Note: Original input is directly concatenated to the features computed by the network.	130
6.11	Hierarchical representation of urban point clouds from Paris data (expert knowledge). First level is composed of 7 nodes: rideable, unrideable, building-like, other, vegetation (14), vertical and movable. Second level represents the leafs of the tree: 1 road, 2 road-line; 3 ground, 4 terrain, 5 sidewalk; 6 building, 7 wall, 8 fence, 9 bridge; 10 rail-track, 11 guard-rail, 12 water, 13 static; 15 pole, 16 traffic-sign, 17 traffic-light; 18 pedestrian, 19 vehicle.	139
6.12	Comparison of predicted labels in Soufflot 0 varying the loss function. The model trained with hierarchical loss can discriminate top vegetation from building. Other models get systematically confused between top vegetation and building class.	142

List of Tables

2.1	Class rates in CARLA data from Paris-CARLA-3D dataset.	33
2.2	Class rates in Paris data from Paris-CARLA-3D dataset.	34
3.1	PSNR and MSE. DCL vs GCL	55
3.2	Percentage of pixels in first four GCL using 10 hemispheres with all images.	56
3.3	Metrics of obtained Gaussians in BSD500 dataset varying spatial weight of our proposed image representation and SLIC algorithm [13].	62
3.4	Results in BSD500 dataset varying spatial weight of our proposed image representation and SLIC algorithm [13]. R: Recall, P: Precision, DE: Density, CO: Compactness and US: Undersegmentation error	62
3.5	Simplification of six HSI using proposed approaches. Error measured in: whole image (a) and regions with powders (b). It is observed how the image simplified keeps spectral diversity of the original image. The quality of reconstruction is evaluated by means of PSNR.	67
3.6	PSNR (dB) of simplified representation. S : Spectral; S-XY : Spectral and spatial, C : Constrained to same slice, NO-C : Non-constrained to same slice. Rows 1 to 5 are images from outdoors dataset [7] and the last five images from Powders dataset [8]	70
4.1	Rate of pixels above different reflectance thresholds by channel in visible range.	84
4.2	Rate of pixels above different reflectance thresholds by channel in NIR range.	85
5.1	Binary segmentation in MNIST dataset with batch size of one. In all cases, U-net architecture with three levels of depth and varying the number of filters in the first layer.	110
5.2	Mean IoU in multiclass segmentation on MNIST.	110
5.3	Accuracy in validation set on Aerial images. First column indicates the used loss function: BCE, Focal BCE, Dice score and power Jaccard.	112
5.4	IoU by class and mean IoU in validation set of Cityscapes [4].	112
5.5	Performance in test set of CVPPP dataset [5] with different loss functions. Each column corresponds to different loss function as follows: binary crossentropy (BCE), Focal BCE (F-BCE), Dice, Tversky with $\alpha = 0.1$ (TV1) and $\alpha = 0.5$ (TV5), classical Jaccard and power Jaccard with $p = 1.2, 1.4, 1.6, 1.8, 2.0$	114
5.6	Performance in low slice from SHREC'20 dataset with different loss functions.	114
5.7	Performance in high slice from SHREC'20 dataset with different loss functions.	114

6.1	Intersection over Union by class in test set of Semantic KITTI dataset.	126
6.2	Class rates population in Street 3D [10] dataset.	128
6.3	Intersection over Union by class in test set.	131
6.4	Results on test set in semantic segmentation task using KPConv architecture. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.	133
6.5	Results on test set in semantic segmentation task using PointNet++ architecture. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.	134
6.6	Results in transfer learning for semantic segmentation task using KPConv architecture on our dataset Paris-CARLA-3D. Results are mIoU in %. Pre-training was done using urban towns from CARLA. <i>no-training</i> : model pretrained with CARLA data, without training in Paris data; <i>no-frozen parameters</i> : model pretrained with CARLA without frozen parameters during training with Paris; <i>no-transfer</i> : model trained since scratch with Paris data.	135
6.7	Results on test set of Paris data in semantic segmentation task with transfer learning from CARLA data. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.	136
6.8	IoU by class of Paris data in semantic segmentation task without most frequent classes. We report only IoU by class. M1: Learning all classes; M2: Without ground-like classes; M3: Without ground-like and building classes. <i>mIoU</i> is the mean IoU of learned classes.	138
6.9	Performance using IoU in Soufflot 0 varying loss function and using hierarchical-based approach. First column displays class names; second column, rate of points by class; CE is categorical crossentropy; CE + Jac 2, states for the sum of categorical crossentropy and power Jaccard with $p = 2$; Jac. 2 is power Jaccard with $p = 2$; Hierarc. is the hierarchical loss with Jac 2 as <i>base loss</i>	140
6.10	Performance using IoU in Soufflot 3 varying loss function and using hierarchical-based approach. First column displays class names; second column, rate of points by class; CE is categorical crossentropy; CE + Jac 2, states for the sum of categorical crossentropy and power Jaccard with $p = 2$; Jac. 2 is power Jaccard with $p = 2$; Hierarc. is the hierarchical loss with Jac 2 as <i>base loss</i>	141
6.11	Results on test set in instance segmentation task. SM: Segment Matching, PQ: Instance Quality, <i>mIoU</i> : Mean IoU	144
6.12	IoU in test set of Paris data with different training scenarios: P_1) 13 classes; P_2) 14 classes; P_3) All classes; P_4) Hierarchical learning. In each scenario, two models were trained with RGB features (W) and without RGB features (NO). Bold text is the highest value by class by test point cloud (S_0 and S_3).	146

Chapter 1

Introduction

1.1 Résumé

Dans ce chapitre nous présentons le contexte et la motivation de cette thèse encadrée dans le projet REPLICA FUI 24. Nous présentons également les objectifs de cette recherche. Les principales contributions et les publications scientifiques sont énoncées à la fin de ce chapitre.

1.2 Motivation

The integration of computer vision and machine learning techniques has considerably increased over the last few years. The democratization of the sensors and the vast increase of the computing power of machines have permitted the development of applications using automatic methods for daily life tasks such as public transport planning, email classification, banking, personal finances, social networks, among others.

One of the applications that have attracted the scientific community's attention since the beginning of 20th century is the autonomous vehicle [20]. The proposed approaches have evolved together with the existing technology: starting with a radio-controlled vehicle equipped with an antenna in 1920's, passing to a system controlled by the electromagnetic fields generated by circuits installed on the highway in 1930's, then including first proximity sensors to the vehicle to detect electronic devices embedded on the road in 1960's. However, it was only until the decade of 1980 that Mercedes-Benz in Germany and Defense Advanced Research Projects Agency (DARPA) in the United States with VITS (for **V**ision **T**ask **S**equencer) [21], included cameras to the vehicle in order to provide a vision-guided system. From this point on, the use of cameras and their later integration with automatic algorithms exhibited a significant improvement of autonomous vehicles' capabilities.

Nowadays, the autonomous vehicle is still a very active research topic. The industrial and scientific communities have joined efforts to propose solutions from different domains such as sensors, control systems, mapping, navigation, and testing. As it is presented by [22], the implementation of self-driving vehicles in coming years still requires much work to do. One of the most recurrent issues to tackle before integrating autonomous vehicles into public transportation is the safety evaluation. It implies studying enough scenarios with all possible configurations that can be found in the real

world. However, in practical terms, it would mean to ride thousands or even millions, kilometers to cover all possible traffic configurations and weather conditions.

An alternative to evaluate the algorithms without spatial and timing constraints is to use simulation tools. The first learning-based solution for the self-driving car was proposed in 1998 and used a neural network trained with synthetic data [23]. At that moment, synthetic data was chosen as an alternative to overcome the difficulties of acquiring real-world data of the road with a camera. The obtained results demonstrated that the use of synthetic data could be helpful to train a model that will be evaluated later in real-world situations.

The use of synthetic data in machine learning algorithms allows tackling the common problem of insufficient labeled data. For some applications, data can be even created *on-the-fly* during training, reducing the required storage to save massive datasets [24]. In order to have reliable models for real-world applications, it is necessary to have as realistic as possible synthetic data. In the context of the autonomous vehicle, synthetic data has been used to generate image datasets such as Virtual KITTI [25], Synthia [26], and FCAV [27]. Additionally, some open-source platforms such as Apollo [28] permit the creation of new data, if required. Obtained results provide essential findings of the capabilities of synthetic data to improve models' performances for a variety of automatic tasks such as semantic and panoptic segmentation, object tracking and classification, and scene completion. However, existing platforms and datasets are not realistic enough to be directly used to test autonomous vehicles for real-life applications.

To overcome the existing gap between simulation environments and real-world conditions in autonomous vehicle testing, the REPLICA project FUI 24 was launched in 2018. This Ph.D. thesis is part of the REPLICA project, and as it is presented with more detail in Section 1.4, we studied the influence of spectral data in mobile mapping applications using 3D point clouds.

1.3 REPLICA project

The REPLICA project seeks to solve three major issues identified in the current state of the simulation platforms for the autonomous vehicle, which are: **i)** Lack of representativity and productivity of available scenarios in static and dynamic contexts; **ii)** Absence of software to automatically reconstruct and describe 3D environments acquired with multiple sensors; **iii)** Low availability of tools to create dynamic scenarios. Additionally, due to the nature of the autonomous car, it is possible it may face scenarios in open world not contemplated during the training stage. The above implies a challenge in the definition of the simulation scenarios and the sequence of tests to be implemented.

Based on the identified shortcomings, the REPLICA project has a general objective to provide new computational tools and integrate them into a simulation platform that allows the development of massive tests in a virtual environment for the autonomous vehicle. This is why, to develop a global solution to the identified problem, the collaborative work of multiple actors from different domains of knowledge is necessary.

Thus, REPLICA FUI 24 project is developed in partnership by companies from different sectors where their knowledge and expertise contributes from different fields and seeks to develop an integral solution to the identified problem. The companies on the project are presented in Fig. 1.1 and listed below:

- Renault (<https://www.renault.fr/>)
- OPTIS (now part of Ansys) (<https://www.ansys.com/>)

- OKTAL (<https://www.oktalsydac.com/>)
- ALL4TEC (<https://www.all4tec.com/>)
- Terra3D (<https://www.terra3d.fr/>)
- PSA Group (<https://www.groupe-psa.com/en/automotive-group/>)
- ARMINES/MINES ParisTech - PSL Research University:
 - CAOR - Center for robotics (<http://caor-mines-paristech.fr/>)
 - CMM - Center for Mathematical Morphology (<https://www.cmm.mines-paristech.fr/>)



Figure 1.1: REPLICA 24 FUI project brings together six industrial partners: OPTIS, Terra3D, Renault, PSA group, ALL4TEC, OKTAL and one academic partner: ARMINES/MINES ParisTech.

1.4 Goals of this thesis

The main objective of this thesis is to provide automatic tools to accelerate the creation of virtual scenarios and improve their realism in the context of the autonomous vehicle. In order to provide an overview of the structure of the conducted research, in Figure 1.2 we present a flowchart together with the proposed contributions in the different domains. The study was divided into three stages as follows:

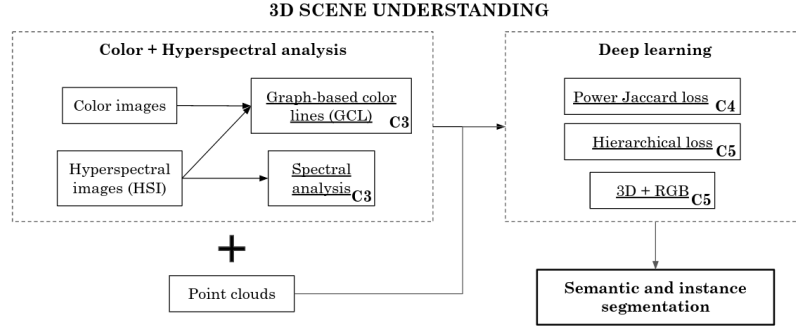


Figure 1.2: Diagram with the main structure of this thesis. The contributions are presented in the boxes with underlined text: Graph-based color lines and analysis of hyperspectral data (Chapter 3); power Jaccard loss (Chapter 5); hierarchical losses and study of RGB influence in semantic segmentation of point clouds (Chapter 6).

Simplified spectrum representation: The spectrum provides information about electromagnetic radiation according to frequency or wavelength. Depending on the range of the spectrum, different information can be extracted. For example, in the range between 400 and 700 nm, it is located the visible spectrum for human beings and provides information about the colors. In another range, between 700 and 1000 nm, known as Near Infrared Range (NIR), invisible information about materials can be extracted. According to the type of application, the representation of the spectrum may vary. In our case, we are interested in building a simplified representation that preserves the spectral diversity of the images. Keeping this diversity allows us to save spectral values even if they are present in little quantities on the image. Our work related to this stage is presented in Chapter 3.

Multispectral data and 3D point clouds in urban scenes: A more detailed description of the spectrum provides additional information about the objects present in the image. In the context of the REPLICA project, we are interested in integrating hyperspectral images (HSI) to 3D point clouds to improve scene description for simulation purposes by recognizing materials of the objects. As a complementary analysis of the influence of spectral data in urban scenes, a comparative study was performed to identify the cases where color information provides useful information that improves the performance of the models. The analysis of spectral information in urban scenes is presented in Chapter 3.

Segmentation of images and point clouds: Segmentation is the process to partition an image, or a point cloud, into homogeneous regions [29]. The criterion to define a region as *homogeneous* changes according to the application: color, texture, or semantic meaning, for example. In this thesis, we studied semantic segmentation of imbalanced datasets and proposed two strategies to tackle it: a novel loss function (Chapter 5) and a hierarchical representation during learning (Chapter 6). To demonstrate the validity of our proposals, we evaluated them with several types of data and a large variety of datasets.

1.5 Thesis outline

Each chapter of this thesis is self-contained, and it is mainly composed following the same structure given by a scientific paper: introduction, state of the art, methodology, results, and discussion. All bibliographic references are presented at the end of the document. The rest of this thesis is organized as follows:

Chapter 2 introduces the different types of data used in this thesis. They are grouped into: images, including grayscale, color and multispectral images from different contexts; 3D point clouds from urban scenes; the data acquired for the REPLICA project including point clouds and multispectral images of urban scenes; and the novel Paris-Carla 3D dataset (PC3D), built in a collaborative work by researchers of CMM and CAOR.

Chapter 3 presents the proposed methodology inspired by [30] in order to represent images using color information. Roughly, the proposed technique simplifies the spectrum of an image by grouping similar pixels according to their spectral information, using Color Lines (CL). The main contribution of our technique is the generalization of lines to piecewise lines using a graph-based approach. From obtained results using only the color, spatial information is included as an additional feature to favor more compact and spatially connected regions in the images. The proposed technique was extended to multispectral and hyperspectral images. The methodology for color images was presented in the 20th International Conference on Image Analysis and Processing (ICIAP, 2019) [31]. The second part of this chapter comprises an analysis of acquired multispectral data in the context of the REPLICA project.

Chapter 5 studies loss functions and their importance during training in deep learning methods for semantic segmentation tasks. The loss function has a crucial role because it permits quantifying the error of the models during training with a single value and improves the performance. A short description of the main stages to train a Deep Learning model is presented. Then, we introduce our proposed Power Jaccard loss [32] and evaluate it with different image datasets, including grayscale and color images, as well as point clouds projections. The contributions of this chapter were published in: 1) The Computer & Graphics journal and 2) The 16th International Conference on Computer Vision Theory and Applications (VISAPP, 2021).

Chapter 6 presents our proposal for semantic segmentation of 3D point clouds for urban scenes. The content covers different aspects of point cloud segmentation, such as different data representations (2D projections or raw 3D), learning methods to segment urban scenes, and the common issue of class imbalance in urban scenes. In the second part of the chapter, we present a comparative study about the influence of color features under different training scenarios. The results obtained in this chapter provide an interesting discussion about the importance of spectral information in point cloud segmentation for urban scenes. The main contributions of this chapter are: proposal of a hierarchical loss for semantic segmentation; the new PC3D dataset, submitted to the International Conference on 3D Vision (3DV, 2021). We contributed to the dataset in three aspects: a benchmark of semantic and instance segmentation tasks; creation of the evaluation protocol of semantic and instance segmentation tasks; manual labeling of semantic classes and instances.

Chapter 7 discusses the main advantages and drawbacks of proposed methodologies in this thesis. Also, we present conclusions and future work.

1.6 Associated publications

This thesis led to the following publications:

1. Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., Goulette, F., & Marcotegui, B. (2019). A graph-based color lines model for image analysis. In ICIAP 2019 [31].
2. Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., Goulette, F., & Marcotegui, B. (2019). Semantization of point clouds using multiscale features. In French-German Doctoral Workshop, Kaiserslautern, Germany, Oct 2019 (Link to program and slides: https://people.cmm.minesparis.psl.eu/research/duque/colloque_2019/).
3. Ku, T., Veltkamp, R. C., Boom, B., Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., ... & Gangisetty, S. (2020). SHREC 2020: 3D point cloud semantic segmentation for street scenes. In Computers & Graphics 2020, volume 93 [10].
4. Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., Goulette, F., & Marcotegui, B. (2020). On power losses for semantic segmentation. In French-German Doctoral Workshop, Virtual meeting, Oct 2020 (Link to program and slides: https://people.cmm.minesparis.psl.eu/research/duque/colloque_2020/).
5. Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., Goulette, F., & Marcotegui, B. (2020). Preliminary results on 3D Point-Cloud Classification through data augmentation for highly imbalanced dataset. In 43ème journée ISS France, MINES ParisTech 2020 (Link to program and slides: https://people.cmm.minesparis.psl.eu/research/duque/ISS_43_2020/).
6. Duque-Arias, D., Velasco-Forero, S., Deschaud, J. E., Goulette, F., Serna, A., Decencièrre, E. & Marcotegui, B (2021). On power Jaccard losses for semantic segmentation. In VISAPP 2021 [32].
7. Deschaud, J. E., Duque-Arias, D., Richa, J.P., Velasco-Forero, S., Marcotegui, B. and Goulette, F. (2021). Paris-CARLA-3D: an Outdoor Point Cloud Dataset for Challenging Tasks in 3D (**Submitted**). In International Conference on 3D Vision (3DV 2021).

Chapter 2

Data used in this thesis

2.1 Résumé

Dans ce chapitre, nous décrivons les données utilisées dans la thèse. Nous introduisons les bases de données sélectionnées pour évaluer nos contributions, aussi bien les images et les nuages de points 3D pour différentes applications. Dans le cadre de ce travail, des nouvelles données ont été produites. D'une part, des données hyperspectrales acquises pour le projet REPLICA. D'autre part, le nouveau jeu de données Paris-CARLA 3D proposé, construit dans le cadre d'un travail collaboratif avec des chercheurs du laboratoire CAOR. Cette nouvelle base de données est composée de nuages de points colorés des scènes urbaines réelles et synthétiques.

2.2 Introduction

As it is described in Chapter 1, one of the main objectives of this work is to analyze the influence of spectral information in the automatic segmentation of point clouds. It involves the analysis of geometrical data provided by 3D point clouds and the spectral information present in color, multispectral, and hyperspectral images. This chapter introduces the types of data used in this thesis and chosen datasets to evaluate our proposals in the following chapters.

The content of the chapter is divided into four sections. In the first section (Section 2.3), we introduce images and selected datasets to evaluate our proposed algorithms (Section 2.3). We worked with grayscale and color images from different contexts, including urban scenes, written digits, and aerial data. We also include multispectral and hyperspectral images and datasets acquired in outdoor and indoor environments.

The second section (Section 2.4) introduces 3D point clouds and selected datasets to perform semantic segmentation tasks. In the context of the autonomous vehicle, we worked with datasets from urban environments. We introduce selected 2D projections from 3D point clouds, used during our participation in the SHREC'20 challenge, presented in Chapter 6.

The third section (Section 2.5) describes the data acquired for the REPLICA project. First, we introduce the acquisition system and include a description of the sensors. Then, we bring up the acquisitions conducted during this thesis, including point clouds, images, and hyperspectral images. We highlight that during this thesis, a joint work between the Center of Mathematical Morphology

(CMM) and the Center for Robotics (CAOR), from MINES Paristech, was carried out to identify issues on acquisitions.

The last section (Section 2.6) presents the novel Paris-Carla 3D dataset, built in a collaborative work by researchers of CMM and CAOR. This dataset is composed of colored point clouds, acquired with the system described in Section 2.5 and synthetic point clouds, generated with the CARLA simulator. This dataset was developed to perform challenging 3D tasks, including semantic segmentation, panoptic segmentation, and scene completion.

2.3 Images

In the development of this thesis, we worked with images from several sources and scenarios including outdoor environments [2], videos under several conditions [1], hand-written digits from well known MNIST dataset, among others. In the following section, we briefly describe chosen datasets and the motivation that guided us to select them. Also, we indicate the sections where the reader can find experimental results with each dataset.

2.3.1 Datasets

Images dataset were selected to evaluate proposed algorithms for images in two stages of this work. The first stage is color representation by means of Graph-based Color Lines model [31], presented in Chapter 3. In this case, we evaluated our proposed approach with different tasks: image representation, the capacity of generalization in videos, and over-segmentation.

The second stage is about the influence of loss function in image segmentation tasks. We proposed a novel Power Jaccard loss [32], introduced in Chapter 5. We analyzed the influence of our proposal with different types of images under different scenarios. Also, we studied the performance of 3D point clouds by using 2D projections.

DAVIS dataset [1] It contains 50 videos in order to perform object detection tasks. Each video was captured at $24fps$ and has a Full HD 1080p spatial resolution. The annotation of each video is provided as a binary mask indicating the presence (or not) of at least one object to be separated from the background. The different types of objects in the dataset can be grouped on: humans, animals, vehicles, and objects. Also, additional annotations for each video are provided to represent specific situations such as fast motion and occluded background. Figure 2.1 shows a frame from snowboard video from DAVIS dataset.

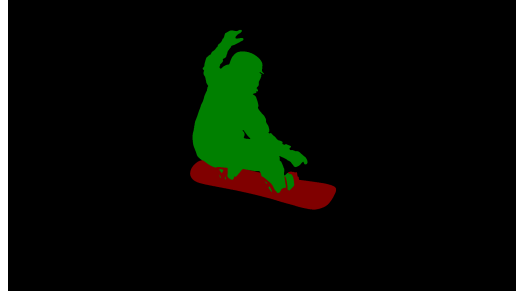
This dataset was selected to evaluate the *capacity of generalization* [33] of the proposed method to build image representations. The experiments performed using color videos are presented in Section 3.4.2.

Spanish cities from Labelme dataset [2] It is composed of images from Spanish cities, derived from the Labelme dataset [34]. It provides 14.871 images for training and 279 for testing. It has been used for very different tasks, including image segmentation, image retrieval, and superpixel evaluation. Figure 2.2 shows two color images from the dataset.

This dataset was selected to evaluate the simplified representation of color images proposed in Chapter 3. It was chosen because it comprises outdoor scenes with uncontrolled light conditions, and we were particularly interested in this type of case. Obtained results with these images are presented in Section 3.4.3.



(a) Frame from snowboard sequence



(b) Ground truth with two labeled objects.

Figure 2.1: Example of color image and its corresponding ground truth from DAVIS dataset [1]



Figure 2.2: Example of two images from Barcelona dataset [2]

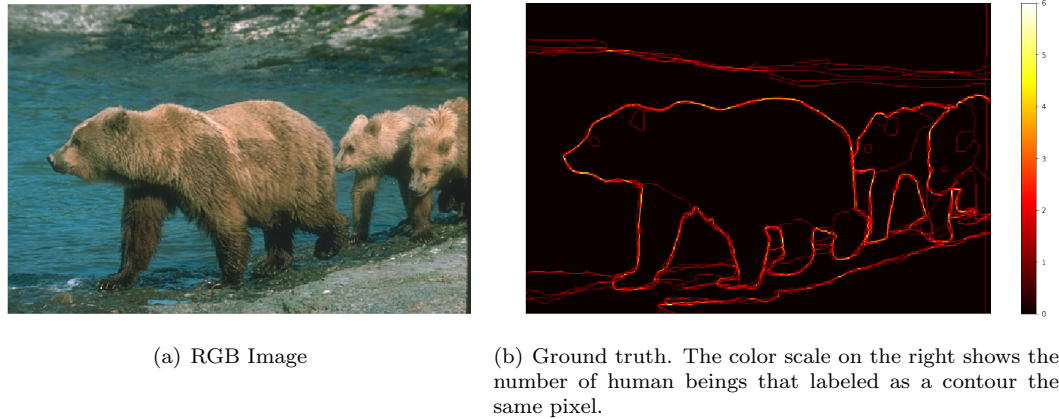


Figure 2.3: Example of color image and its corresponding ground truth from BSD-500 dataset [3]

Berkeley segmentation (BSD 500) [3] It is composed of 500 color images acquired under several natural conditions. It is commonly used to perform image segmentation and boundary detection tasks. One of the characteristics of this dataset is that the ground truth of each image is obtained from boundaries marked by several human beings. It allows to have a soft boundary map where according to the application, the detection of contours changes. Figure 2.3 shows an example of a color image from the training set and its corresponding ground truth map.

This dataset was selected to perform over-segmentation tasks using the proposed methodology, including spatial information in image representation. Obtained results are presented in Section 3.4.3.

MNIST It is a grayscale dataset containing images of 28x28 pixels to perform classification tasks with digits of ten classes. In our case, we created an MNIST-based dataset inspired by [35] in order to perform semantic segmentation tasks. We built the dataset to evaluate our proposals for semantic segmentation tasks by using well-known images.

We randomly selected 1.4K images per class and built a pixel-wise ground truth (14K images in total). This choice was done in order to generate a small dataset to test proposed algorithms. Figure 2.4 displays an example of the dataset with one instance per class. The dataset is available at <http://www.cmm.mines-paristech.fr/~duque/MNIST-segmentation/>. We provide a snippet of code to load images and ground truth.

The built dataset was used to perform binary segmentation and multiclass segmentation tasks, using the proposed power loss function introduced in Chapter 5. Obtained results with this dataset are presented in Section 5.7.

Cityscapes It is a color image dataset developed to perform semantic segmentation tasks of urban street scenes [4]. It has been acquired in 50 European cities, most of them in Germany, under several weather conditions. It is composed of 5K fine annotated images with 30 different classes.

This dataset was selected to perform semantic segmentation in four classes: person, car, road,



Figure 2.4: Images from generated MNIST dataset for segmentation

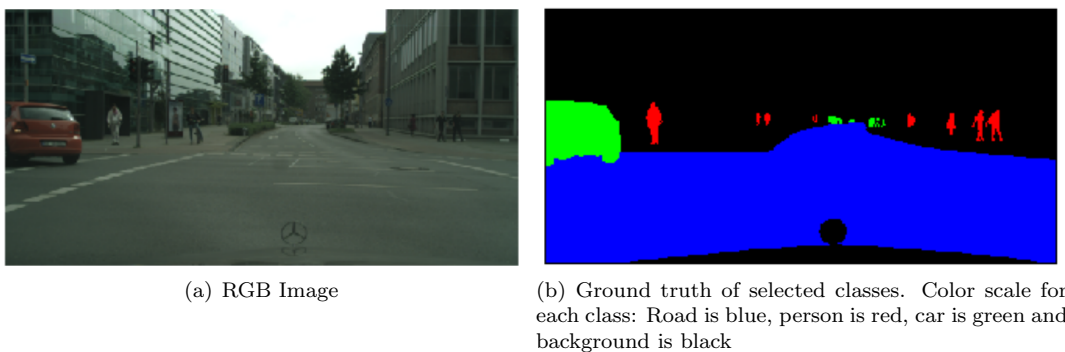


Figure 2.5: Selected classes of example image from Cityscapes dataset [4]

and background. They were chosen in the context of autonomous driving. Figure 2.5 shows a color image and its corresponding ground-truth of selected classes. Experiments carried out with this dataset are presented in Section 5.7.

Plants phenotyping dataset [5] It is a dataset composed of RGB images of two types of rosette plants (*Arabidopsis* and tobacco) and constructed to study leaves growing and plants phenotyping. Tobacco images were acquired using a camera with a single plant in its field of view. The dataset is composed of 810 images with manually annotated labels to perform leaf counting. Other tasks can also be performed, such as plants classification.

In our case, we selected this dataset in order to perform a binary segmentation task and identify plant and *no-plant* pixels. Figure 2.6 displays an example of RGB image, original ground truth to perform leaves counting, and ground truth to perform binary segmentation. The experiments performed using plant images are presented in Section 3.4.2

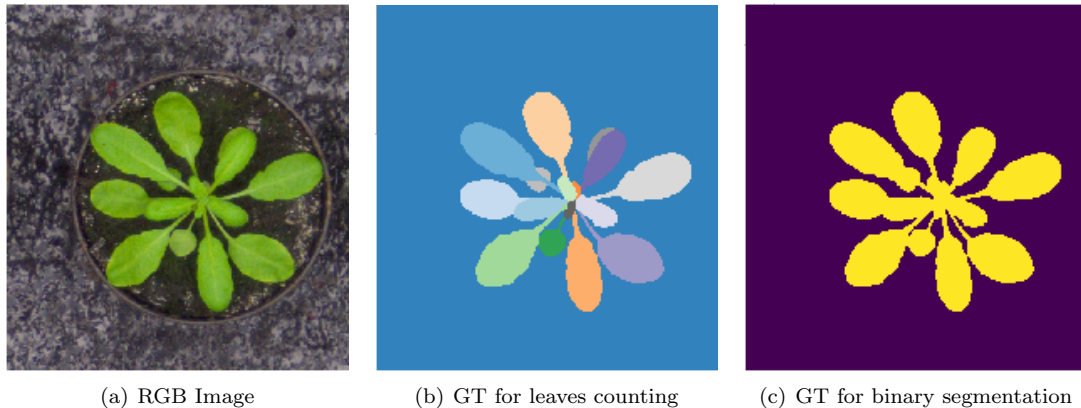


Figure 2.6: Example image from CVPPP dataset [5] and its ground truth to perform leaves counting and binary segmentation.

Aerial dataset from Toronto University [6] It is a dataset introduced by [6] and composed of RGB aerial images. It is divided in three subsets as follows: 1) Massachusetts road dataset; 2) Massachusetts building data set; 3) Buffalo roads dataset. In this work, we performed experiments with the first two datasets presented before. Obtained results with this data can be found in Chapter 5.

The first subset, the Massachusetts road dataset, consists of 1171 images from Massachusetts of 1500x1500 pixels, covering an area of 2.25 square kilometers. Images are pixel-wise labeled with “road” and “non-road” classes, to perform binary segmentation tasks. The data was randomly split by [6] into 1108 images for training, 14 for validation, and 49 for testing. The second subset, the Massachusetts building data set, consists of 151 images of the Boston area of 1500x1500 pixels, covering 2.25 square kilometers. Images are pixel-wise labeled with “building” and “non-building” classes. The data was randomly split by [6] into 137 images for training, 10 for validation, and 4 for testing. Figure 2.7 presents two images from the dataset and their corresponding ground truth.

2.3.2 Multispectral/Hyperspectral Images

Definition

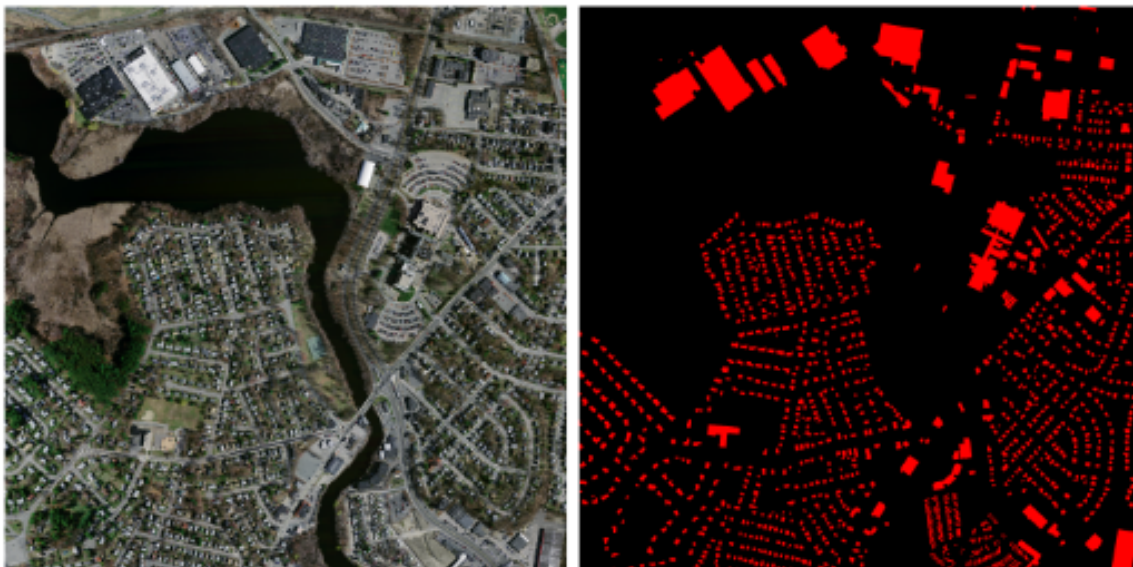
A multispectral/hyperspectral image is composed of a set of n bands acquired across the electromagnetic spectrum. Each pixel is represented as a vector of dimension n , with values obtained at different wavelengths from reflected energy according to the type of material in the scene [36]. According to reflectance response over the electromagnetic spectrum, materials in the scene can be identified utilizing its spectral signature [37].

Indeed, a hyperspectral image I could be represented as a cube with dimension h, w, n , where h is image height, w is image width, and n is the number of acquired channels, as shown in Figure 2.8.

Multispectral Images (MSI) and Hyperspectral Images (HSI) are used in applications where a more detailed description of spectral information is required. In general, they provide from a few dozens of contiguous spectral bands, in the case of MSI, up to several hundred in the case of HSI



(a) Image from Massachusetts roads dataset



(b) Image from Massachusetts building dataset

Figure 2.7: Example of aerial images from Toronto University dataset [6](color and ground truth). The dataset is composed of three sub-datasets in order to perform different binary segmentation tasks. We selected two of them to perform some experiments in our work: the Massachusetts roads dataset and the Massachusetts building dataset.

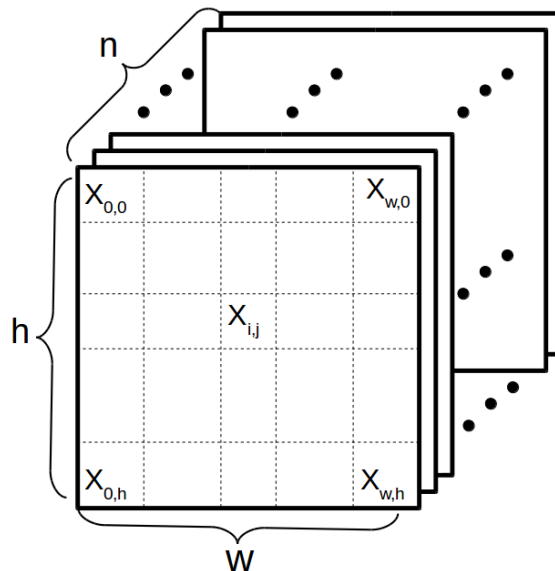


Figure 2.8: Representation of a hyperspectral image

[38]. The most popular and studied applications of this data are related to remote sensing [39]. Some of them are: diagnosis of earth surface [40], archeology [41], mineral mapping [42], agriculture and forestry [43], identification and mapping of wetland vegetation [44], among others.

Thanks to the great advances of this technology and the growing availability of several datasets, their use has been extended to other applications. For example, [45] describes an algorithm to estimate the state of the paint on a building using HSI; [46] presents a review of advances in quality and safety inspection of fruits and vegetables using HSI; estimation of temperatures using terahertz frequencies [47]; detection of freezing parameters in different environments and temperatures [48], monitoring cured meat during drying process [49], among others.

Datasets

This section introduces selected datasets to perform some experiments using proposed approaches with MSI and HSI. They were chosen to evaluate the representation of higher dimensional data through the proposed GCL model for color images. As it is detailed in Section 3.5, our proposition seeks to preserve the spectral diversity of images. Additionally to datasets presented in this section, we also worked with a set of HSI and colored point clouds acquired by the Center for Robotics (CAOR) from PSL Research University in Paris. The data acquired in the context of the REPLICIA project is presented in Section 2.5.

University of the Negev [7] It is composed of 100 images acquired from outdoor environments in several scenes, such as urban, suburban, rural, indoor, and plant-life. Images have 31 bands between 400 *nm.* and 700 *nm.* Figure 2.9 shows some images of the dataset. It can be seen how uncontrolled light conditions strongly affect the color of objects such as trees.



Figure 2.9: Images from University of the Negev dataset [7] with RGB colors computed from hyperspectral data.

This dataset was used to evaluate the quality of image reconstruction using the extension of original GCL for hyperspectral images introduced in Section 3.5. Images in this dataset are challenging due to the strong influence of light in reflectance values over the different bands.

Powders dataset [8] It is composed of images of 100 types of powders acquired in controlled conditions with several backgrounds in a laboratory. Each image has 961 channels, including RGB, Near Infrared (NIR), and Short Wave Infrared (SWIR) information. Figure 2.10 shows some images of the dataset. It can be seen that, in general, powders are not distinguishable in all spectral bands.

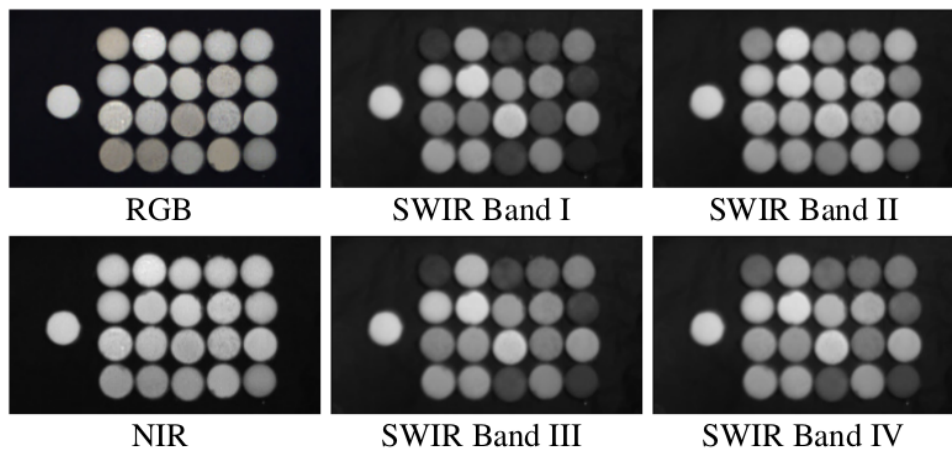


Figure 2.10: Images from powders dataset [8].

This dataset was used to evaluate the quality of image reconstruction introduced in Chapter 3. The main difference from experiments performed using University of the Negev dataset [7], in this case, we evaluated the quality of reconstruction in the region where the powders are located. It allowed us to confirm that the proposed model preserves the spectral diversity of MSI. Obtained results are presented in Section 3.5.2.

2.4 Point clouds

Points clouds are a type of data commonly used to model 3D objects. In the beginning, their use was strongly constrained due to the elevated cost of sensors. The development of low-cost sensors such as Kinect of Microsoft Xbox 360 and the publication of some libraries such as Point Cloud Library (PCL) [50], motivated their use in a wider range of applications. As presented by [51], the use of point clouds over the years has expanded significantly, including 3D model reconstruction, virtual reality as well as the widespread autonomous driving application.

In this thesis, we worked with different types of point clouds, mainly from urban scenes, to perform segmentation tasks. As it is presented with more detail in Chapter 6, we evaluated the influence of spectral information from color and HS images in several semantic segmentation scenarios. In the following sections, a more formal definition is introduced as well as the selected datasets to evaluate our proposed algorithms.

2.4.1 Definition

Points clouds are given by a set of 3D points acquired, in general, with a Light Detection and Ranging (LiDAR) laser. In certain cases, geometrical information can be integrated with additional features such as color or intensity [19]. A point cloud is then represented by a geometrical part given by $P \in \mathbb{R}^{N \times 3}$ and the features $F \in \mathbb{R}^{N \times D}$. Figure 2.11 presents an example of a point cloud with color information and intensity. It can be seen how different objects are easier to recognize when additional features are included, compared to point clouds with pure geometric information.

2.4.2 Datasets

In this section, we introduce selected datasets to perform point cloud segmentation with proposed methods of Chapter 6. In some cases, we worked with 2D projections, and in others, we did it directly with 3D data. In Section 2.4.3 we present the different types of point clouds projections. Moreover, in Section 2.4.4 is introduced the concept of neighborhoods and their importance in point clouds.

For a complete survey of point clouds datasets, we refer the reader to [52].

Semantic KITTI [9]

It is a large-scale dataset composed of 22 sequences of urban scenes under very different traffic conditions, acquired in Germany. The data contains 23.201 fully annotated scans for training and 20.351 for testing. Each point cloud has about 150K points with geometrical information (x, y, z) , intensity values, and it is manually labeled ground truth. This ground truth is given from a set of 20 classes in the context of the autonomous vehicle. Figure 2.12 displays an example of a point cloud of Semantic KITTI. This dataset was selected in order to perform semantic segmentation using spherical projections. Results obtained using this dataset are presented in Section 6.3.2.

Street 3D [10]

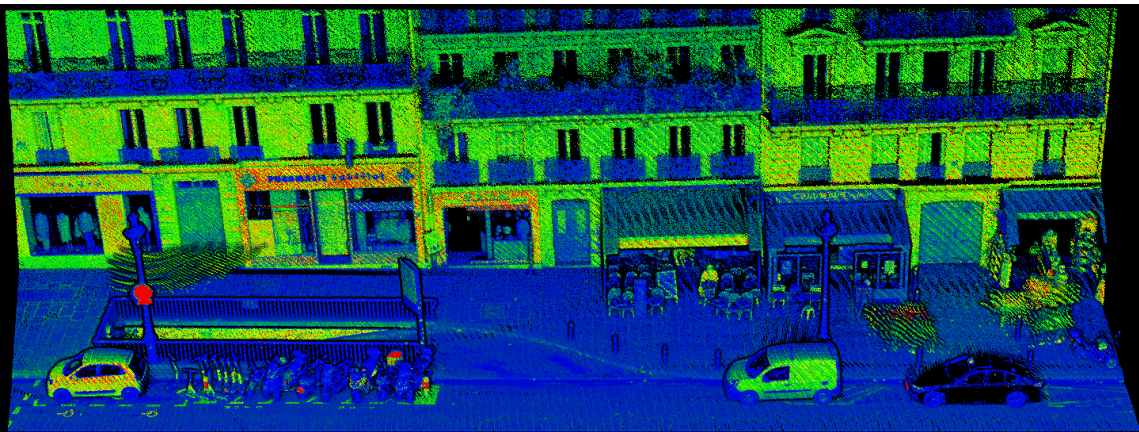
The dataset was provided by the organizers of the SHREC'20 challenge on the track "3D Point Cloud Semantic Segmentation for Street Scenes". It is composed of 80 point clouds of urban scenes, divided into 60 point clouds for the training set and 20 for the test set. Each point cloud has about



(a) Point cloud without features

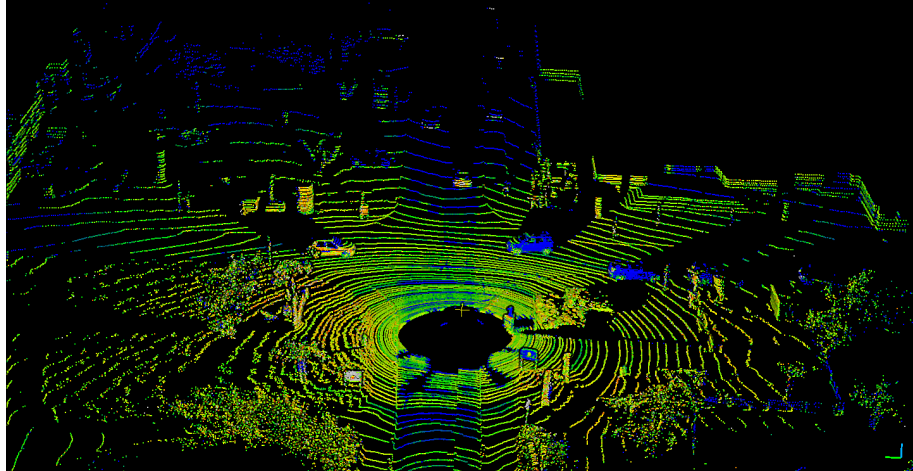


(b) Point cloud with color information

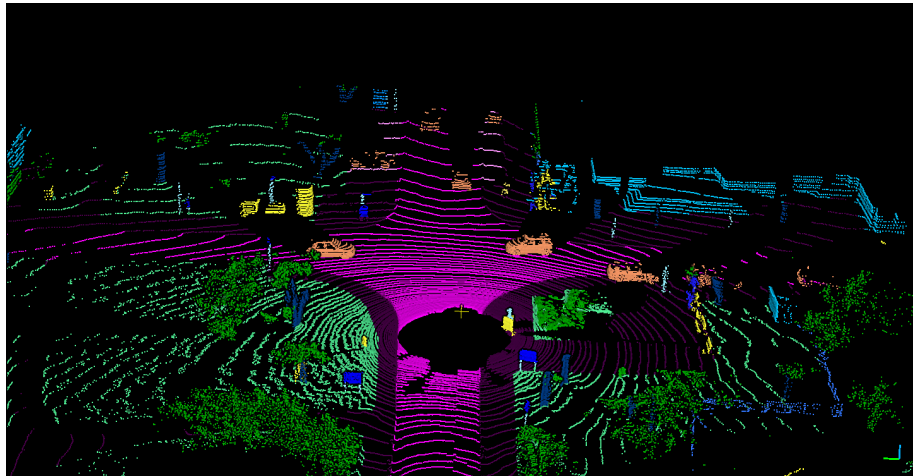


(c) Point cloud with intensity values: Low reflectance is labeled with blue, intermediate with yellow/green and high with orange/red

Figure 2.11: Example of 3D point cloud with color and intensity information from rue Soufflot, in Paris.



(a) Point cloud with intensity values: Low reflectance is labeled with blue, intermediate with yellow/green and high with orange/red



(b) Point cloud with ground truth

Figure 2.12: Example of a 3D point cloud from Semantic KITTI dataset [9]. Color scale of most visible classes: Road is purple, traffic signs are blue, cars are orange, vegetation is green, low vegetation is light green and buildings are blue.

2 million points, and it was manually labeled with the following classes: *building*, *car*, *ground*, *pole* and *vegetation*. Figure 2.13 displays an example of a point cloud of Street city dataset. Obtained results using Street 3D dataset are presented in Section 6.3.2.

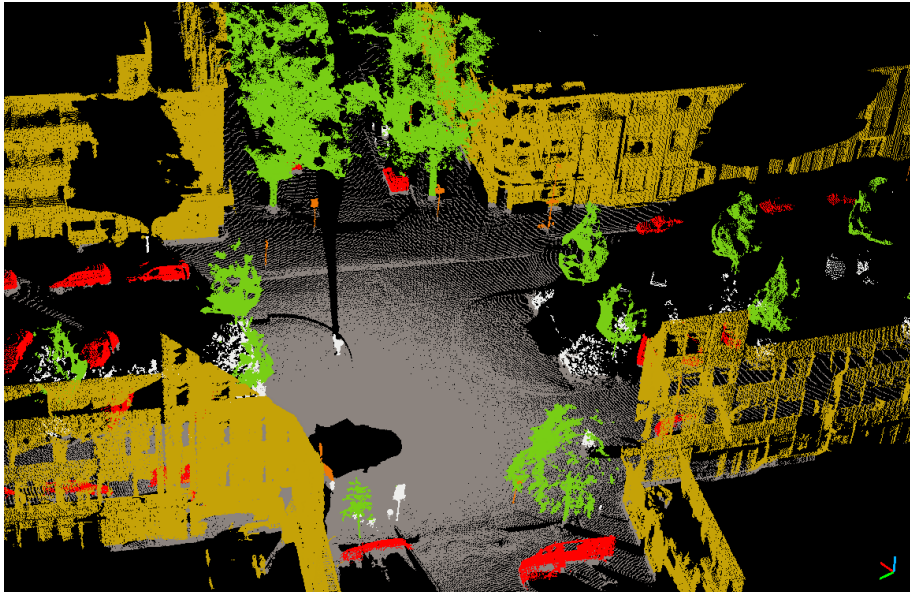


Figure 2.13: Example of 3D point cloud from Street Scenes dataset [10]. Color scale: buildings are yellow, cars are red, ground is gray, vegetation is green, poles are orange and unlabeled points are white.

2.4.3 2D projections

The research community has widely used projection of 3D point clouds over the years [53]. The main goal is to build of a 2D representation of point clouds preserving the most relevant 3D information. In the context of point clouds from urban scenes, two of the most commonly used projections are: *top view projection*, representation of 3D data as it could be seen by a “bird” from the sky; *spherical projection*, most frequently used in Velodyne scanners, it seeks to build a representation from the sensor point of view. Figure 2.14 displays top view and spherical projections of a point cloud from Semantic KITTI.

The representation of 3D data with 2D projections has some characteristics such as:

- Dense and structured representation of sparse data.
- Reduction of data dimensionality and as, an indirect consequence, a reduction in computational cost.
- Build images of different features from 3D data, e.g. intensity, elevation, accumulation.

Additionally to already introduced features, one of the main reasons to use 2D projections is that they allow one to take advantage of the maturity of existing machine learning algorithms with

images and apply them to perform tasks with 3D point clouds. However, 2D projections have some drawbacks such as: 1) Some geometrical relationships between 3D points can be lost; 2) Some objects can be occluded, especially in Top view projections; 3) Objects are deformed and their size can be altered in spherical projections.

During this thesis, we worked with 2D projections with two datasets to perform semantic segmentation tasks: spherical projections with Semantic KITTI (Section 6.3.2); spherical and top view projections during SHREC'20 challenge (Section 6.3.2). Additionally, we studied the recent panoptic segmentation with Paris-Carla 3D dataset and BEV projections. In the following sections, we describe the top view and spherical projections.

Top-view projection Top-view projection, also known as Bird Eye View (BEV) projection, is widely used by research community. It allows to project onto 2D plane 3D points while preserving all geometric information except their height. The transformation matrix of a 3D point cloud to an orthogonal projection is given by:

$$T_{orth} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

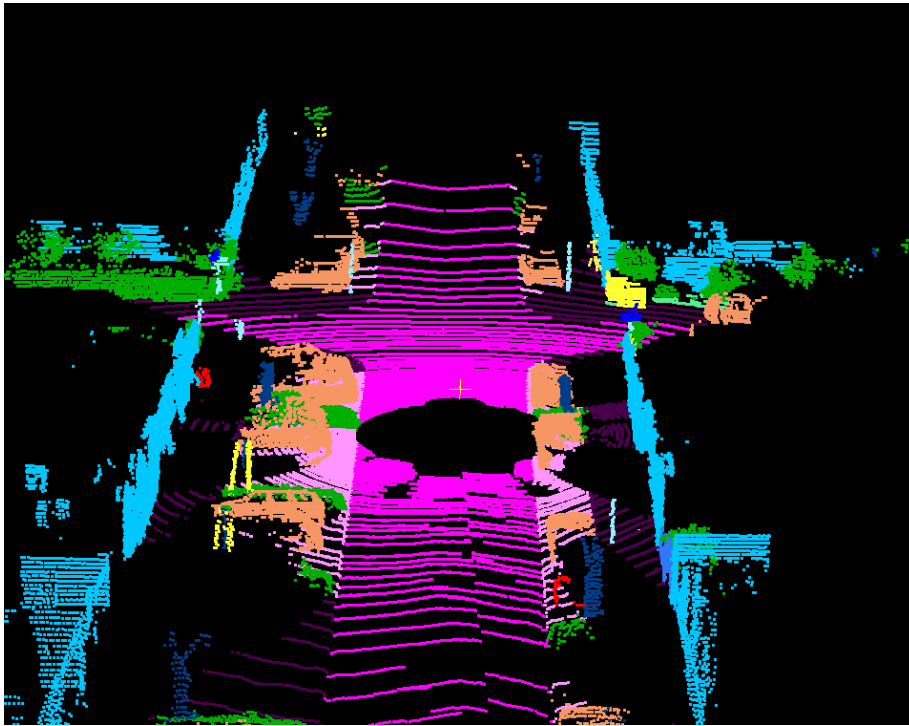
Every 3D point has its corresponding coordinates in the xy plane. In some cases, more than one point can be projected onto the same (u, v) coordinates of the obtained plane. It implies that occlusions may occur, and objects can be hidden if they are located below other ones. For example, in point clouds from urban scenes, it is common to have trees or bus stops with pedestrians below them. The presence of occlusions can be tackled by performing projections at different heights, as proposed by [54].

Top view projection can be easily computed employing Equation 2.1. It is required to define a voxel size in order to reduce data sparsity and computational cost. The selected size will impact the resolution of the projection. This is the only parameter to perform a top view projection from a 3D point cloud and it must be adjusted according to the application. If the voxel is too small, the projection will have many holes due to data sparsity. If it is too big, it will combine onto the same pixel points from different objects.

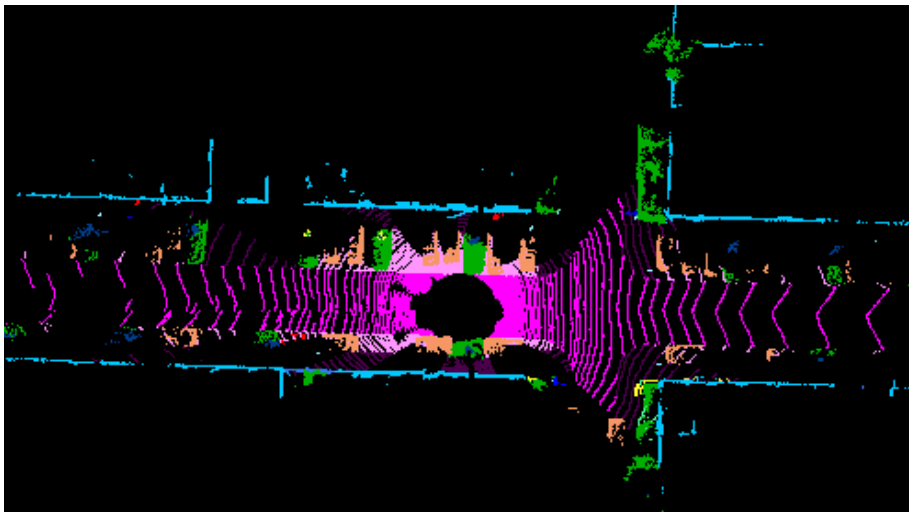
In general, every feature available in a 3D point cloud can be projected to a top view projection. Some of most commonly used feature projections for ground detection in urban scenes are [54]:

- I_{max} : Stores maximal elevation of points projected onto the same pixel.
- I_{min} : Stores minimal elevation of points projected onto the same pixel.
- $I_{\Delta h}$: It is defined as $I_{max} - I_{min}$
- I_{acc} : Stores the number of points projected onto the same pixel.

These projections rely only in geometrical features and are frequently used to create the Digital Elevation Model (DEM) of the point cloud.



(a) 3D point cloud



(b) Top view projection colored with voxel size of 20 cm per pixel



(c) Spherical projection colored with 1032x64 resolution

Figure 2.14: Projections of point cloud from Semantic KITTI dataset coloured with ground truth labels.

Front-view projection This type of projection seeks to build a representation from the LiDAR point of view. The scanner perceives the environment through a set of n laser layers. Each layer has an emitter and a receiver unit and turns 360 degrees around the Z -axis. The whole point cloud is obtained by assembling acquisitions of all layers. As a manner of example, Figure 2.15 displays an illustrative representation of a Velodyne with $n = 4$ and projection of the hollow cylinder onto a plane.

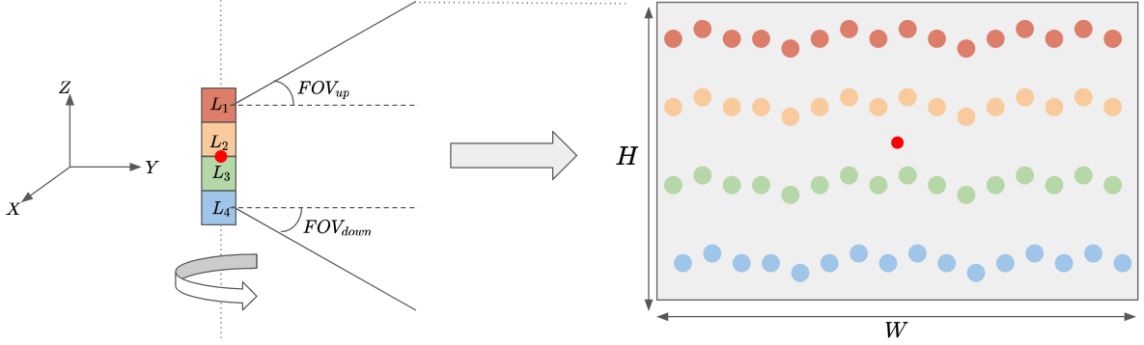


Figure 2.15: Simplified representation of Velodyne with $n = 4$. FOV_{top} and FOV_{down} are the maximum and minimum angles of the field of view. According to the number of layers and angle values, dimension of the rectangle changes. In most common commercial scanners, usually $n = 32, 64, 128$ and FOV angles are between 15 and 30 degrees. Recently, new scanners with $n = 8, 16$ are available

From Figure 2.15, it can be seen that the origin of the sensor, marked with a red circle on the left, corresponds to the center of the projection of the cylinder on the right. Also, it can be noted that the dimension of 2D projection is associated with the number of layers n and FOV angles.

The goal of building a front view representation of a point cloud is to calculate pixel coordinates for every 3D point. By using spherical coordinates, each 3D point can be represented by three numbers: radial distance to the scanner (r); a polar angle (θ), also known as *azimuth*; and a zenith angle (ϕ).

The spherical coordinates of a given point p_i with known geometrical coordinates (x_i, y_i, z_i) are defined as follows:

$$r = \sqrt{x_i^2 + y_i^2 + z_i^2} \quad (2.2)$$

$$\theta = \arcsin\left(\frac{z_i}{r}\right) \quad (2.3)$$

$$\phi = \arctan\left(\frac{y_i}{x_i}\right) \quad (2.4)$$

The calculation of pixel coordinates from the spherical coordinates of each point requires normalization and scaling stages. The first must be performed to define the size of the projected image: height depends on the number of layers of the scanner, and width depends on scanner resolution.

For example, in the case of Velodyne HDL 64-E composed of 64 layers and a horizontal resolution of 0.35 degrees, the expected size of the projected image will be 64x1028 pixels.

Normalization of θ and ϕ is defined as follows:

$$\theta_n = \frac{FOV_{up} - \theta}{FOV_{up} - FOV_{down}} \quad (2.5)$$

$$\phi_n = \frac{\phi + \pi}{2\pi} \quad (2.6)$$

Calculation of pixel coordinates (u_i, v_i) is defined as follows:

$$u_i = H \cdot \frac{1 - (\theta_n + FOV_{down})}{FOV_{up} + FOV_{down}} \quad (2.7)$$

$$v_i = W \cdot \frac{\phi/\pi + 1}{2} \quad (2.8)$$

Applying previous equations, a 3D point P_i with euclidean coordinates (x_i, y_i, z_i) can be projected onto an image in the coordinates (u_i, v_i) . These equations must be applied to every point of the point cloud to obtain a front view projection.

Similarly, as stated before for top view projections, every feature from the point cloud can be directly projected onto a front view projection. Additionally, more features such as depth (r , from Equation 2.2) from projected images, can be calculated.

Front view projections are very useful to visualize point clouds from the scanner point of view. It prevents object occlusion as occurs in top view projections. However, the geometry of objects and their dimensions can be very distorted. Additionally, due to the nature of spherical projections, images are circular. It implies that the left and the right borders of the image are actually connected. These issues must be dealt with attention for specific applications, especially in object detection tasks.

This type of projection is commonly used in deep learning architectures such as SqueezeNet [55], SalsaNet [56], Rangenet++ [57] and PointSeg [58]. In this thesis, we worked with this type of projection in order to perform semantic segmentation of point clouds from urban scenes from Semantic KITTI [9] and during the competition of SHREC'20 challenge [10]. In Chapter 6 is presented a detailed description of our proposal using spherical and Bird Eye View (BEV) projections.

2.4.4 Neighborhood

As presented in last Section, 3D point clouds are characterized as being unordered, unstructured and sparse data. It represents a challenge for automatic tasks from several points of view, including high computational cost and feature extraction. These two issues are usually linked by the notion of *neighborhood*, a well-known concept in the image processing field.

In image processing, a neighborhood can be directly identified thanks to the structured representation of the data. It allows performing 2D convolutions with no restriction about image resolution. During the last years, Convolutional Neural Networks (CNN) have dominated the field of image segmentation [59, 60]. However, in 3D point clouds, the neighborhood can not be directly established. Neighborhood representation can be mainly divided into two groups: voxelization methods and 3D raw methods.

Voxelization The first approaches to define neighborhoods on point clouds were based on voxelization methods [61, 62, 63]. Voxels are a type of geometry analogous to pixels of 2D images for 3D data. They allow to order data and build a structured grid representation of point clouds as with 2D images. Other drawbacks appear associated with a computational cost that rapidly increases according to voxels number and the loss of resolution due to the discretization step. In Figure 2.16 is presented an example of voxelization with different voxel size of a vehicle. It can be seen that reducing voxel size improves spatial resolution and increases the number of voxels that represent the original point cloud. This augmentation also increases the computational cost of the algorithms. For the interested reader, [64] provides a detailed revision of methods and applications of voxel-based representation in point clouds.

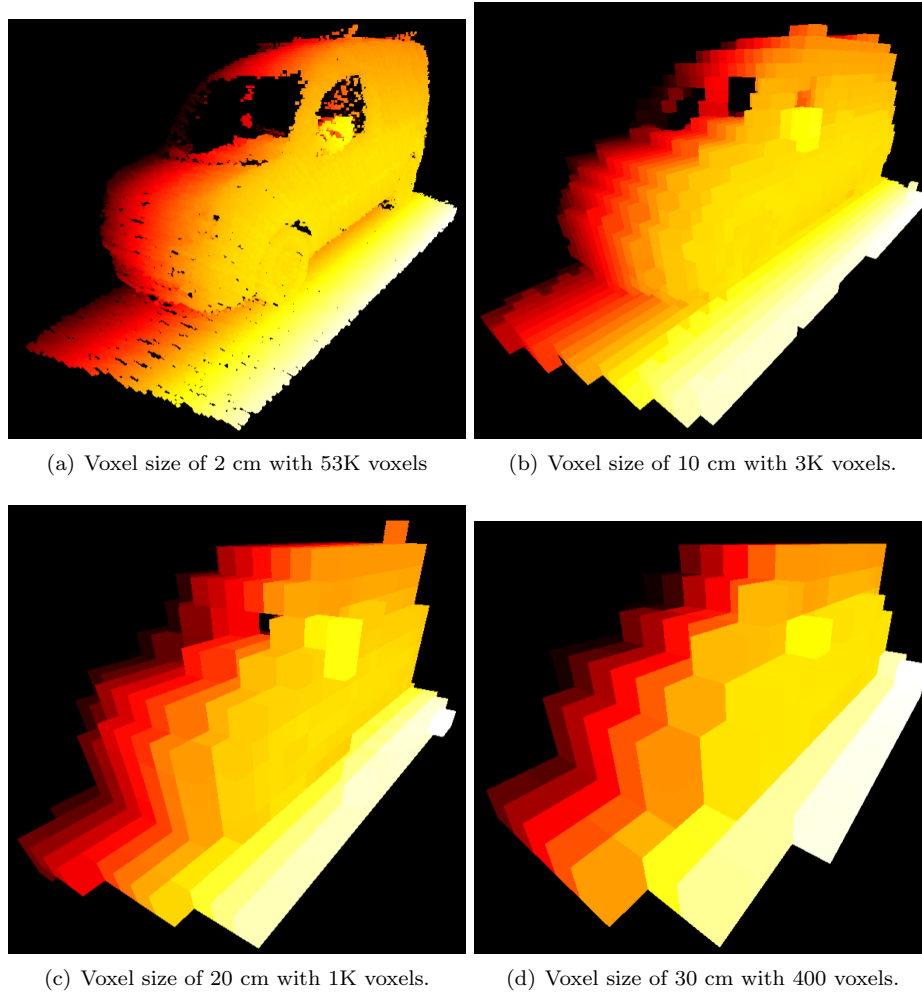


Figure 2.16: Voxelization of a car varying voxel size. Color scale corresponds to x coordinate of voxels center.

3D raw More recently, a second set of approaches have been proposed in order to directly process raw 3D points. Its main goal is to use every available point and extract most of the geometrical information from 3D data. On the one hand, it prevents loss of spatial resolution due to voxelization. On the other hand, a strategy to select consistent neighborhoods must be defined.

Several alternatives have been proposed in order to calculate neighborhoods directly from 3D point clouds. They can be grouped into: 1) Spherical neighborhoods; 2) K nearest neighbors (KNN); 3) Other approaches. In the first category, the neighborhood of a point p_i contains all points located closer than a given radius r . It allows to group pixels by a geometrical criterion. Even though its main limitation is scale selection, leading to inconsistent neighborhoods for objects of different sizes.

In the second category, points are grouped based on their closest k neighbors. As previously mentioned for spherical neighborhoods, the main issue is the scale selection, especially in outdoor point clouds where data density varies. It implies that the selection of neighborhoods based on a radius or based on the closest points will always rely on the selected scale. This is a limitation that may affect smaller objects. As a manner of example, Figure 2.17 displays neighborhoods of two objects: spherical neighborhood with $r = 1$ m. (see Figure 2.17(a)) and K-NN with $k = 2000$ (see Figure 2.17(b)). One may observe how neighborhood changes due to density variations, especially at the pole (yellow). In the case of the car, it is an object with a more regular density because it was closest to the LiDAR scanner. This simple case demonstrates the relevance and complexity of neighborhood selection to extract relevant features for point cloud classification.

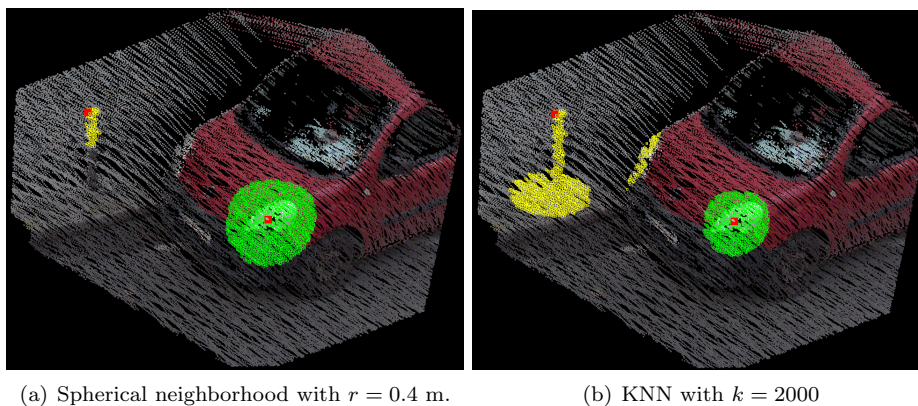


Figure 2.17: Neighborhood of two points (in red) from an outdoor scene point cloud.

The third category groups different strategies, most of them inspired by KNN and spherical neighborhoods, for different objects. As an additional effect, this group can also be interpreted as a subsampling strategy to reduce data volume while preserving most geometrical information. For the interested reader, a detailed description of subsampling methods and grouping strategies for neighborhoods in 3D point clouds is presented by [65].

As a part of this research, we studied semantic segmentation of point clouds for urban scenes (Chapter 6). One of the main challenges of working with this type of data was the strong density variations that affect neighborhood calculation. We studied three subsampling strategies: random sampling, farthest point sampling, and multiscale spherical neighbors, commonly used in some deep

learning architectures.

Random sampling: It is a technique that assigns the same probability to be chosen for every point of the set. It is expected that a randomly chosen sample is an unbiased representation of the total population. However, in point clouds where data density has considerable variations, the sampled representation may not contain enough points of different objects from the scene. PointCNN [66], and ShellNet [67] are two architectures that seek to order points to perform invariant convolutions in point clouds.

Farthest Point Sampling (FPS): It was initially proposed by [68] to subsample 2D images and then generalized for higher-dimensional data, as point clouds. It makes part of *uniform sampling* algorithms. The main idea behind FPS is to repeatedly place the next sample point in the least-known area of the sampling domain. The process to sample a point cloud P_{init} into a set S composed of N points using FPS is described in Algorithm 1.

Input : P_{init}, N

Select random point p_1 from P_{init} ;

Add p_1 to list of sampled points S ;

Remove p_1 from P_{init} ;

for $n \leftarrow 1$ **to** N **do**

 Find farthest point p_f of P_{init} from S ;

 Add p_f to S ;

 Remove p_f from P_{init} ;

end

Output: S sampled set with N points

Algorithm 1: Farthest point sampling method

This subsampling strategy is used in PointNet++[18], VoteNet [69] and PVN3d [70]. Then, neighborhood selection can be performed by radius search (PointNet++) or by KNN calculation.

Multiscale spherical neighborhoods [71]: This method is inspired by [72] defining neighborhoods in 3D using spheres instead of K-closest points. This approach deals with two of the most common issues due to density variations in point clouds: 1) Regions with too many points may considerably increase the computational cost; 2) Regions with too few points will not be discriminant enough. For the first issue, they provide a parameter ρ to define the maximum number of points in a neighborhood. Moreover, for the second issue, its influence is restricted by the use of multiple spherical scales. This method is used in [19] and has been successfully implemented in several types of tasks with point clouds of indoor and outdoor scenes.

2.5 REPLICA data

One of the objectives of the REPLICA project is to improve the realism and robustness of virtual simulations for autonomous driving. A way to do it, is to include multispectral/hyperspectral imagery to provide a more detailed description of the spectrum. Thus, during acquisitions, not only data from LiDAR sensors and color cameras must be acquired but also the images provided by hyperspectral cameras.

The use of Mobile Mapping Systems (MMS) to acquire point clouds with color information is not recent, and several datasets have been published as presented in Section 2.4. However,

there are not previously reported works, fusing hyperspectral data and point clouds [73, 74] for autonomous driving applications. As a part of the REPLICIA project, a team composed of researchers of MINES Paristech developed the acquisition system to generate a dataset with colored point cloud and hyperspectral images. Using several sources to build the dataset was a challenging task from several points of view: calibration for each scanner and between different scanners, synchronization, filtering, and data fusion. The expected output of the acquisition system was a set of colored multispectral point clouds from urban scenes.

In this section, we introduce the scanning system (Section 2.5.1) as well as the acquisitions performed by researchers of MINES Paristech during the development of this thesis (Section 2.5.2). We highlight that during this thesis, several studies were performed with each acquisition to evaluate the influence of spectral information in semantic segmentation tasks. These studies are detailed in Chapter 3.

2.5.1 System description

The acquisition system is integrated into a platform called L3D2 [75], commonly used to perform 3D mobile mapping. The L3D2 is a Citroën Jumper equipped with a GPS (Novatel FlexPak 6), an IMU (IxseaPHINS in LANDINS mode), and a Velodyne HDL-32E LiDAR mounted at the back of the truck with an angle of 30 degrees between the axis of rotation and the horizontal. Additionally, it is equipped with a Ladybug 360° spherical camera to acquire color images. This platform was also used to build the Paris Lille 3D dataset [76] and the Paris Carla 3D dataset, introduced in Section 6.3.2. In Figure 2.18 is presented the L3D2 platform as well as some of the reference frames of its sensors.

Figure 2.19 displays the set of sensors mounted on the roof of the L3D2 vehicle. In this picture, both hyperspectral cameras (HS02 and HS03), Ladybug camera, Velodyne LiDAR, and the antenna of the GPS can be seen.

Point clouds

The platform has a Velodyne HDL-32E LiDAR, composed of 32 lasers aligned from $+10.67^\circ$ to -30.67° vertical field of view and a full 360° horizontal field of view. The HDL-32E generates a point cloud of up to 700K points per revolution with a range of up to 100 m and a typical accuracy of ± 2 cm. This kind of sensor is widely used in the field of MMS. A more detailed description of this sensor can be found on [77].

As it is commonly used in MMS, a dense point cloud is built from several scans acquired with the car in movement. In an ideal case, the dense representation could be built based on the localization of the 3D scanner [78]. Due to the imprecision of georeferenced positions as well as the overlapping between consecutive scans, an additional SLAM (Simultaneous Localization and Mapping) task must be carried out [79]. In our case, dense point clouds were built by means of Implicit Moving Least Squares (IMLS) - SLAM method, proposed by [80].

Color images

The L3D2 is equipped with a Ladybug spherical camera that acquires images at 360° . The system includes six cameras that cover around 90% of the sphere. Figure 2.20 illustrates a raw acquisition at Rue Soufflot, in Paris. From these images, it can be seen that due to the wide angles lens, it is

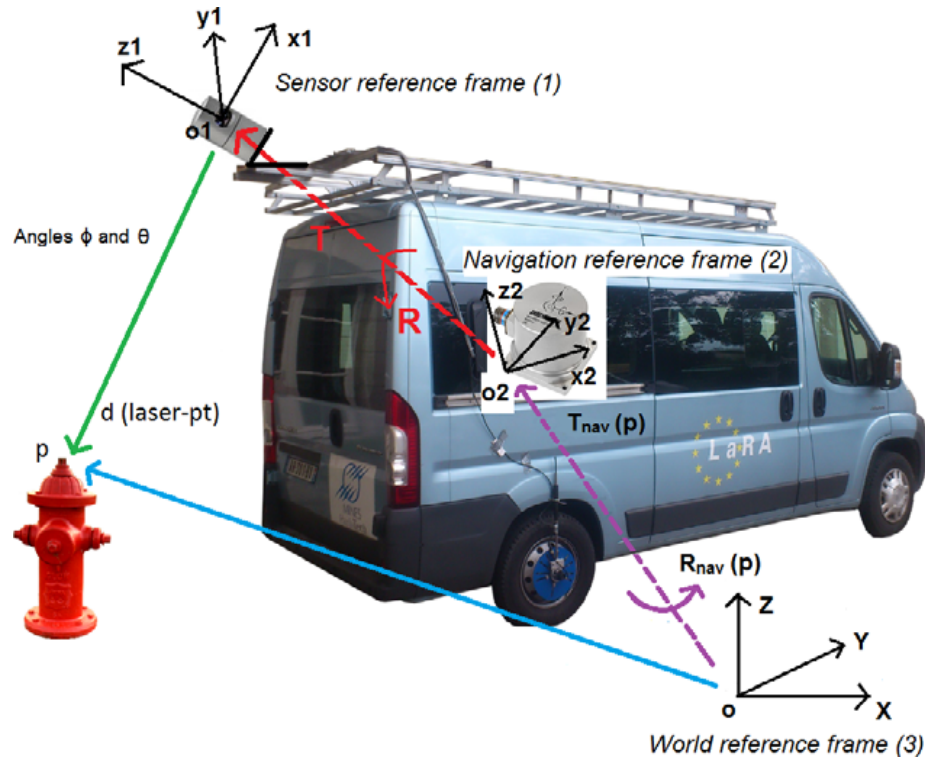


Figure 2.18: L3D2 platform equipped with several sensors: LiDAR sensor, location system and a set of color and multispectral cameras. Each sensor has a reference frame with respect to is acquired the data.

required to perform distortion modeling. More details about the Ladybug5 FLIR system as well as the distortion modeling are presented in [81].

This camera was attached to the top side of the L3D2 vehicle and synchronized with the GPS unit to take a picture with a frequency of 1 Hertz. After some experiments, it was identified that acquisitions were affected by unexpected stops due to traffic lights and traffic jams. This issue was tackled with a second condition to take a picture only if the vehicle had advanced 1 meter since the previous picture.

Hyperspectral cameras

In the context of the REPLICA project, two hyperspectral cameras were included in the platform. These cameras are Photonfocus HS02 and Photonfocus HS03. Both of them have the same working principle of the mosaic so that each pixel of the raw image filters a particular band of the spectrum. The HS02 camera has a mosaic of 5×5 (25 channels) and HS03 of 4×4 (16 channels). As a manner of example, Fig 2.21 displays a raw image acquired with HS02 camera in CAOR laboratory with the mosaic pattern stated before.

The rest of this section about hyperspectral cameras describes the selected references and their



Figure 2.19: Sensors mounted on the roof of L3D2 vehicle: 1) Velodyne HDL-32E LiDAR; 2) Ladybug camera; 3) Antenna of GPS; 4) Hyperspectral cameras HS02 and HS03.

main characteristics. Then, the calibration stage is conducted before every acquisition is presented. Finally, the postprocessing step in order to demosaic raw images is introduced.

HS02 The HS02 camera, with a full commercial reference MV1-D2048x1088-HS02-96-G2, is a hyperspectral camera covering the Near Infrared (NIR) spectrum between 600 nm. to 975 nm. This type of camera has been used in other works, such as remote sensing applications [82, 83] and fruits classification in controlled light conditions [84].

The resolution of acquired raw images is 2048*1088 pixels with 25 spectral bands. Due to the mosaic representation of the images and the need for a postprocessing stage, the size of the output images is reduced to 409x217 pixels and 23 spectral bands.

This camera was attached to the left side of the vehicle, as shows the Figure 2.19. The same conditions to trigger a photo were defined: frequency of 1 Herz and only if the vehicle has moved 1 meter since the previous picture. The exposure time was fixed to 5 ms.

HS03 The HS03 camera, with a full commercial reference MV1-D2048x1088-HS03-96-G2, is a hyperspectral camera covering the visible range of the electromagnetic spectrum from 470 nm. to 630 nm. This camera was also used in fruits classification in controlled light conditions [84]. Other works with this camera are available, such as a comparison with other hyperspectral sensors to prevent laser dazzle projection [85] and review about the evolution of cameras in UAV (Unmanned Aerial Vehicles) spectroscopy [86].

The resolution of raw images is the same as the HS02 camera (2048*1088 pixels), but in this case, they contain 16 channels. As stated before, the unmosaicing stage reduces the dimensions to 512x256 pixels and 13 channels.

This camera was attached next to the HS02 camera, on the left side of the L3D2 vehicle. The same conditions to trigger a photo were defined: frequency of 1 Herz and only if the vehicle has moved 1 meter since the previous picture. The exposure time was fixed to 5 ms.

Calibration The calibration of both hyperspectral cameras was performed through the privative software *Hyperspectral SDK*, provided by Photonfocus. As stated in the documentation of the manufacturer [87], the calibration process of each camera is divided into:

1. Use the camera's XML correction file and select the rejection filters used to apply the spectral correction. The manufacturer provides this file.
2. Get a dark reference at the image's exposure time: the dark reference image is captured with the lens closed and only contains bias noise.
3. Get a white reference image: an image of a white uniform surface that reflects the light uniformly and is used to maximize the camera's dynamic range and characterize the system. In practice, this reference image was a white target with a reflectance close to 98 %.

Cameras calibration was carried out in two different scenarios: 1) Inside the laboratory under fully controlled light conditions; 2) Outside the laboratory, in the parking of MINES Paristech. In the first stage, the calibration stage allowed the acquisition of hyperspectral images covering the whole dynamic range, as expected. Even though, in the second stage, the variation of light conditions due to uncontrollable events such as clouds displacement strongly affected the quality of the image. Also, as will be described later in this section, the presence of dynamic objects impacted the quality of hyperspectral images.

Due to the nature of *Hyperspectral SDK* software, owned by Photonfocus and only accessible with a paying license, we did not have access to source code.

Postprocessing A postprocessing stage was included in the process to normalize reflectance values and build a full resolution image from mosaic images acquired with the cameras. This step was performed by a research engineer hired by the CAOR laboratory in the REPLICA project based on a recent method proposed by [88].

2.5.2 Acquisitions

The acquisition system provides data from four different sources: LiDAR scanner, Ladybug camera, HS02 camera, and HS03 camera. Each sensor acquires information from an environment of different nature, and the task of assembling them to generate expected output was a continuous

and incremental task. During this thesis, four acquisitions of urban scenes from Paris were studied: Autumn 2019, Summer 2020, End-summer 2020, and Autumn 2020. We emphasize the acquisition season because the illumination conditions and the objects present in the scene change between them. Furthermore, all the acquisitions were carried out in the neighborhood between Luxembourg Gardens and Pantheon in Paris. Figure 2.22 shows the area where all acquisitions were done.

In the following sections, we describe the acquisitions performed during this thesis. The collaborative work with researchers of the CAOR laboratory allowed us to identify issues in acquisitions and evaluate alternatives to correct them. These studies are detailed Chapter 3.

Autumn 2019 (V1)

The first data acquisition was performed in December of 2019. It is constituted of four colored point clouds and 90 HS images by camera. Each point cloud is comprised of 10 million points including (x, y, z) coordinates, intensity, and color information. As described before, dense point clouds were generated from a set of successive scans of the L3D2 vehicle equipped with Velodyne sensors using IMLS-SLAM method [80]. The color images were acquired with Ladybug camera and then projected to dense point clouds. Figure 2.23 displays a point cloud from the first acquisition. Each point is colored with reflection intensity and RGB information, respectively. At the time of writing this thesis, the dataset was submitted to Remote Sensing journal (mdpi).

Autumn 2019 (V2)

The data of the first acquisition had some errors in the postprocessing stage of HS images. The error over-saturated reflectance values of the images, as it is described with more detail in Chapter 3. It was corrected by researchers of the CAOR laboratory, and the images acquired in V1 were generated again. It means that the data of V2 contains only HS images.

Summer 2020

It was performed in June 2020. It is constituted by 16 colored point clouds and 547 HS images using the same configuration of the acquisition system described before. The data was acquired at this time of the year to maximize the objects' luminosity: an issue identified during the analysis of previous data acquired in winter.

Static summer 2020

It is composed of three images of the HS03 camera. In contrast with previous ones, this acquisition was performed in a static environment, in the parking of MINES Paristech university, in Paris. The data was acquired to evaluate the influence of exposure time of cameras with respect to the quality of spectral information.

Images were acquired successively with the camera fixed in the same position. Three different exposure times were tested: 5 ms, 50 ms, and 100 ms. Figure 2.24 presents the three images painted in false RGB colors from HS data.

Autumn 2020

It was selected to build Paris Carla 3D dataset, described in Section 2.6. It is composed of six colored point clouds without hyperspectral information. With this last acquisition, we studied

several approaches to evaluate the influence of RGB information to improve the classification of certain classes. Performed experiments with Paris Carla 3D dataset are presented in Section 6.3.2 and include semantic segmentation and panoptic segmentation tasks. The most relevant findings of the influence of color information in point cloud segmentation are described in Chapter 6.

2.6 Paris-Carla 3D dataset

The Paris-Carla 3D (PC3D) dataset was created by MINES Paristech in collaborative work between the CMM and the CAOR. The main objective of building PC3D was to provide a challenging and voluminous dataset to test and improve existing methods in outdoors point clouds on different tasks such as semantic segmentation, panoptic segmentation, and scene completion. The paper with PC3D was submitted to the Remote Sensing journal (mdpi).

The dataset is divided into two parts: synthetic point clouds generated with CARLA simulator [89] and point clouds acquired with an MMS including a LiDAR and color cameras in Paris.

2.6.1 CARLA data

CARLA is an open-source simulator [89] that runs over Unreal Engine 4 and is commonly used in autonomous driving applications. It provides an API to configure several variables in the virtual environment such as cameras, LiDAR sensors, weather, among others. CARLA simulator was configured with LadyBug camera and Velodyne HDL32, the same equipment as the experimental platform L3D2, described in Section 2.5.1.

The simulator provides seven towns from urban and rural scenes. Those cities were digitally created and are not a virtual twin of a real-life town. Data's acquisition was performed employing an autonomous vehicle that moves around each city, acquiring point clouds and color images. Then, LiDAR scans are assembled onto a dense point cloud by using the simulator's ground truth trajectory of the scanning system. It enables to project the color information acquired with the camera onto 3D data. From each town, a dense point cloud with 100 million points was generated. Semantic labels for 22 classes and instance labels for two classes (vehicles and pedestrians) are available in CARLA data.

2.6.2 Paris data

Point clouds were acquired close to Luxembourg Gardens in Paris. This zone is very dense and contains many static and dynamic objects that represent a challenge for automatic tasks. Similarly as performed before for previous acquisitions of the REPLICIA project, LiDAR scans are aggregated by using IMLS-SLAM [80] to create dense point clouds. In total, six point clouds including color information were generated, each one is composed of 10 million points. Manual annotation of Paris data was performed by collaborative work between CAOR and CMM laboratories. Semantic labels were given based on CARLA documentation about class descriptions. Instance labels were assigned to vehicle instances.

2.6.3 Dataset properties

PC3D is composed of two different types of point clouds: synthetic and real-world scenes. It permits to have many types of objects belonging to the same class, as it occurs in towns from the real world.

It not only increases the complexity of semantic segmentation tasks but also implies defining a test set that represents most of the classes variability.

The distribution of classes by point cloud is stated in Tables 2.1 and 2.2 for CARLA data and Paris data, respectively. From CARLA data, as occurs in real world scenarios, not every class is present in every town: eleven classes are present in all towns (*road, building, sidewalk, vegetation, vehicles, road-line, fence, pole, static, dynamic, traffic-sign*), three classes in six towns (*unlabeled, wall, pedestrian*), three classes in five towns (*terrain, guard-rail, ground*), two classes in four towns (*bridge, other*), one class in three towns (*water*), two classes in two towns (*traffic-light, rail-track*).

Class	CARLA data							Total
	T_1	T_2	T_3	T_4	T_5	T_6	T_7	
unlabeled	5.81%	2.92%	-	7.64%	0.03%	6.41%	1.79%	3.51%
building	6.75%	22.61%	15.34%	4.45%	16.11%	2.59%	3.32%	10.17%
fence	0.97%	0.63%	0.02%	0.47%	3.81%	1.45%	0.59%	1.13%
other	-	-	-	-	0.05%	0.05%	0.11%	0.03%
pedestrian	0.13%	0.16%	0.05%	0.04%	-	0.05%	0.01%	0.06%
pole	0.56%	0.59%	4.22%	0.76%	0.78%	0.42%	0.29%	1.09%
road-line	0.23%	0.17%	2.91%	1.56%	2.22%	1.34%	1.68%	1.44%
road	47.78%	37.15%	53.09%	52.81%	44.73%	58.03%	42.84%	48.06%
sidewalk	22.54%	17.54%	10.30%	1.68%	10.48%	3.14%	0.44%	9.45%
vegetation	8.69%	10.78%	2.69%	12.82%	4.59%	8.08%	23.14%	10.11%
vehicles	1.66%	3.11%	0.86%	0.46%	3.07%	4.16%	0.86%	2.03%
wall	1.94%	3.63%	1.43%	5.42%	5.28%	3.40%	-	3.02%
traffic-sign	-	0.03%	-	0.06%	0.01%	0.02%	0.08%	0.03%
ground	-	0.02%	0.17%	1.37%	0.29%	0.05%	-	0.27%
bridge	1.66%	-	-	0.72%	6.60%	-	-	1.28%
rail-track	-	-	7.58%	-	0.49%	-	-	1.15%
guard-rail	0.01%	-	-	4.33%	-	1.20%	0.45%	0.86%
traffic-light	0.05%	0.06%	-	-	-	-	-	0.02%
static	0.76%	0.50%	0.32%	0.30%	0.30%	-	-	0.31%
dynamic	0.11%	0.11%	0.10%	0.27%	0.07%	0.05%	0.06%	0.11%
water	0.36%	-	0.04%	-	-	-	0.55%	0.13%
terrain	-	-	0.88%	4.83%	1.08%	9.57%	23.80%	5.74%
# Points	700M							

Table 2.1: Class rates in CARLA data from Paris-CARLA-3D dataset.

Concerning Paris data, classes variability is smaller than in CARLA data. It is a desired (and expected) characteristic of these point clouds because they correspond to the same town and were successively acquired by a Mobile Mapping System. Even though, as it occurs with CARLA towns, not every class is present in every point cloud: twelve classes are present in all point clouds (*road, building, sidewalk, road-line, vehicles, other, unlabeled, static, pole, dynamic, pedestrian, traffic-sign*), three classes in five point clouds (*vegetation, fence, traffic-light*), one class in two point clouds (*terrain*) and six classes in any point cloud (*wall, ground, bridge, rail-track, guard-rail, water*).

We proposed data splits in PC3D in order to perform semantic and panoptic segmentation tasks. Several alternatives were studied due to high variability of classes that constrain the selection of some point clouds as test set:

- **Train/Validation data:** Town 2, 3, 4, 5 and 6; Soufflot 1, 2 and 4.

Class	Paris data						
	S_0	S_1	S_2	S_3	S_4	S_5	Total
unlabeled	0.85%	1.48%	3.94%	3.18%	1.87%	0.94%	2.04%
building	14.89%	18.85%	34.18%	36.56%	33.09%	32.89%	28.41%
fence	2.29%	0.62%	0.71%	0.82%	-	0.42%	0.81%
other	2.05%	3.41%	6.74%	2.15%	2.50%	0.41%	2.88%
pedestrian	0.22%	1.02%	0.64%	1.01%	0.65%	0.65%	0.70%
pole	0.56%	0.94%	0.59%	0.79%	0.73%	1.06%	0.78%
road-line	3.79%	3.68%	2.39%	4.09%	3.53%	3.44%	3.49%
road	40.95%	49.73%	35.03%	37.56%	40.61%	27.53%	38.57%
sidewalk	10.09%	4.19%	7.27%	6.68%	11.85%	29.43%	11.59%
vegetation	18.48%	9.04%	0.11%	0.30%	0.07%	-	4.67%
vehicles	1.26%	1.77%	6.47%	6.47%	3.27%	1.63%	3.48%
wall	-	-	-	-	-	-	-
traffic-sign	0.14%	0.38%	0.13%	0.11%	0.29%	0.05%	0.18%
ground	-	-	-	-	-	-	-
bridge	-	-	-	-	-	-	-
rail-track	-	-	-	-	-	-	-
guard-rail	-	-	-	-	-	-	-
static	2.64%	2.28%	0.26%	0.06%	0.74%	1.52%	1.25%
traffic-light	0.12%	0.16%	0.05%	0.06%	0.05%	-	0.07%
dynamic	0.28%	1.63%	1.49%	0.15%	0.73%	0.04%	0.72%
terrain	1.39%	0.81%	-	-	-	-	0.37%
# Points	60M						

Table 2.2: Class rates in Paris data from Paris-CARLA-3D dataset.

- **Test data:** Town 1 and 7; Soufflot 0 and 3.

We used previous data splits to train and evaluate proposed algorithms for the benchmark presented in the submitted paper of the Paris Carla 3D dataset. Most relevant findings and proposed approaches for semantic and panoptic segmentation tasks are presented in Section 6.3.2.

2.7 Conclusions

In this chapter, we introduced the data types and selected datasets used during this thesis. They can be divided into two main groups: images and point clouds. In the first group, we worked with grayscale and color images (Section 2.3), as well as multispectral and hyperspectral images (Section 2.3.2). In the second group, we worked with 3D point clouds of urban scenes (Section 2.4). We introduced selected datasets and described performed acquisitions in the context of the REPLICa project (Section 2.5). A relevant contribution of this chapter is the proposal of the new Paris Carla 3D dataset (Section 2.6), created in joint work with researchers of CMM and CAOR laboratories. This dataset is currently under review in Remote Sensing journal.

As the reader will find in the following chapters, these datasets were selected to study spectral information in images and point clouds. Firstly, by proposing a way to represent it and analyzing the data acquired from the REPLICa project (Chapter 3); Secondly, by carrying out semantic segmentation tasks in images (Chapter 5) and point clouds (Chapter 6) for different types of ap-

plications; and finally, by analyzing their influence in segmentation of urban street scenes (Chapter 6).

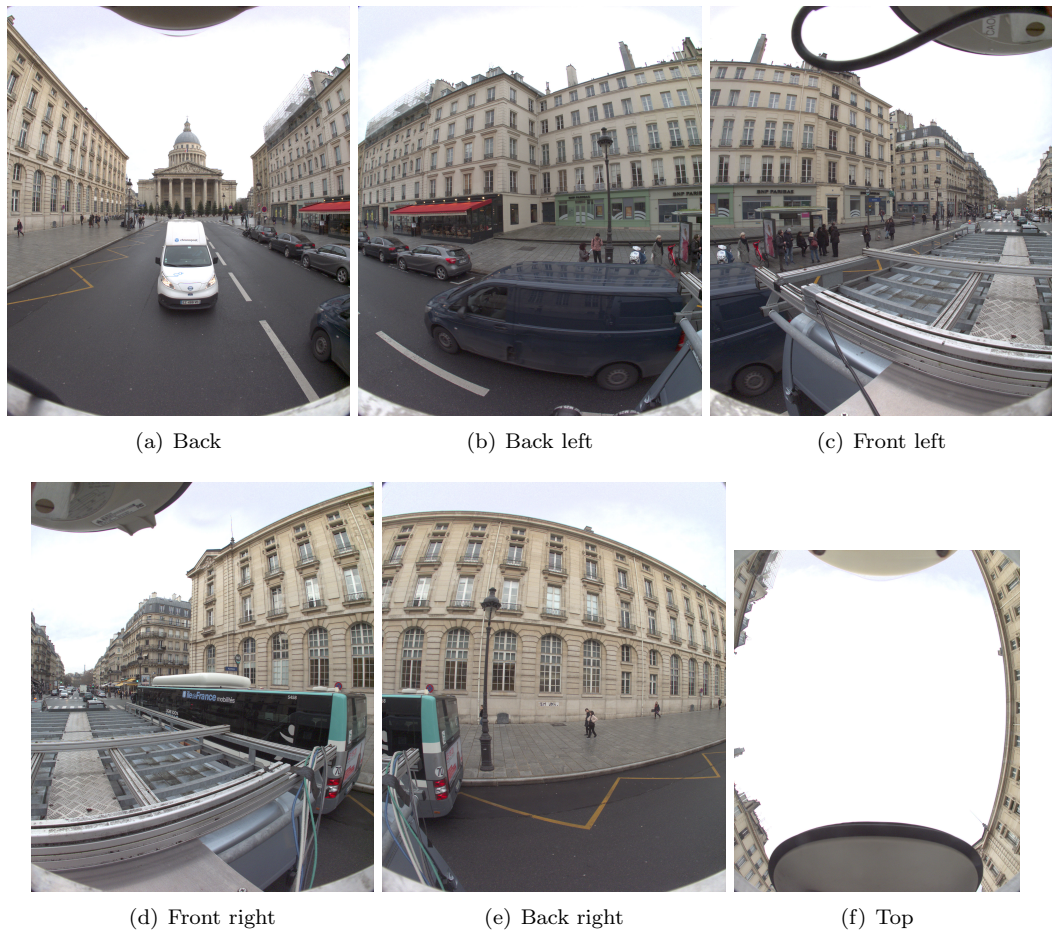
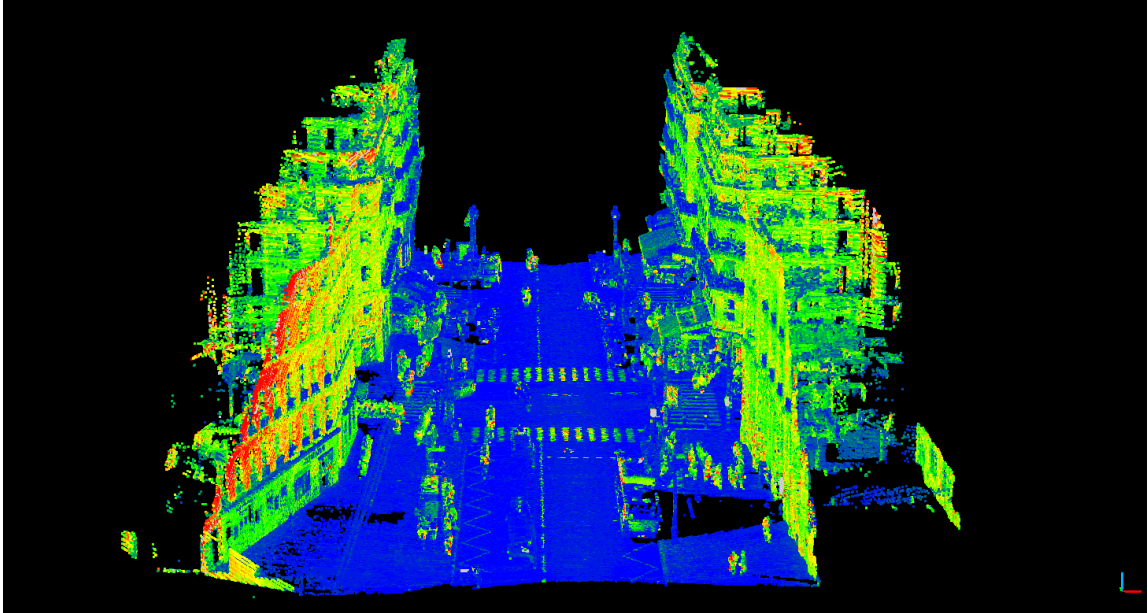
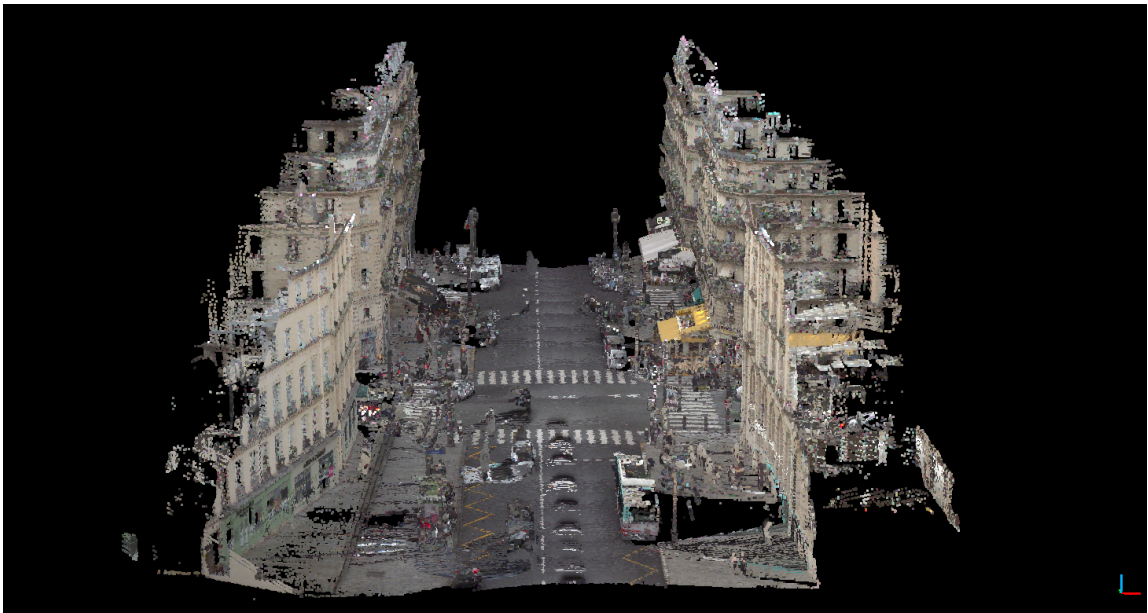


Figure 2.20: Raw color images acquired with LadyBug camera at Rue Soufflot in Paris. Each camera acquires a part of the 360° sphere (about 90 %, according to manufacturer).



(a) Reflection intensity: Low values are blue, intermediate green/yellow and high red.



(b) Color information

Figure 2.23: Dense point cloud from the first acquisition. It contains 10 million points and includes intensity (Fig. 2.23(a)) and color information (Fig. 2.23(b)). Other fields are also included in the file, such as: scan angle, GPS time, sensor position, vertical laser angle, laser index and frame index.



(a) 5 ms of exposure time



(b) 50 ms of exposure time



(c) 100 ms of exposure time

Figure 2.24: Images in false RGB colors from HS03 camera. Three different exposure times were tested in the fourth acquisition: 5 ms, 50 ms and 100 ms.

Chapter 3

Graph based color lines (GCL)

3.1 Résumé

Dans ce chapitre, nous proposons une généralisation de la méthode Color Lines. Les lignes droites sont remplacées par de poli-lignes définies à partir d'un arbre couvrant de poids minimal dans l'espace RGB. Cette nouvelle méthode (Graph-based Color Lines, GCL) permet de simplifier les images couleur en préservant leur diversité spectrale. Nous présentons aussi l'extension de GCL en ajoutant l'information spatiale de l'image pour améliorer la compacité et connectivité des régions. Nous introduisons aussi la méthode GCL pour les images multispectrales (MSI) et hyperspectrales (HSI).

3.2 Introduction

In this chapter, a model is introduced to build a simplified representation of color images and multi-spectral images. It is called Graph-based color lines (GCL), and it represents spectral data through piecewise linear functions. It is inspired by *color lines* [30] method which have demonstrated successful results in various applications such as video compression and segmentation [90], interactive shadows correction [91] and image dehazing [92]. The proposed model considers non-straight lines by computing paths on a graph-based representation. Accordingly, the model is called *Graph-based Color Lines* (GCL), and it is mainly composed of the following three steps: **i)** Compute simplified spectral representation; **ii)** Graph modeling of the color representation; **iii)** Build graph-based color lines. The last step is a graph-based generalization of color lines method [30], based on the spectral representation of images. The proposed GCL method was presented in the 20th International Conference on Image Analysis and Processing (ICIAP) [31].

GCL model permits the identification of more complex geometrical structures in spectral representations than Dynamic Color Lines (DCL) [90] and keeps at the same time the potential to analyze the image. Figure 3.1 shows a comparison between the proposed method against DCL. As a manner of example, four color lines were extracted for both models and the quality of image reconstruction was measured with Peak Signal to Noise Ratio (PSNR). It can be seen how the proposed GCL has a higher PSNR because it represents better spectral diversity of the original image. In Section 3.4 is presented a detailed explanation of the representation of color images using

GCL.

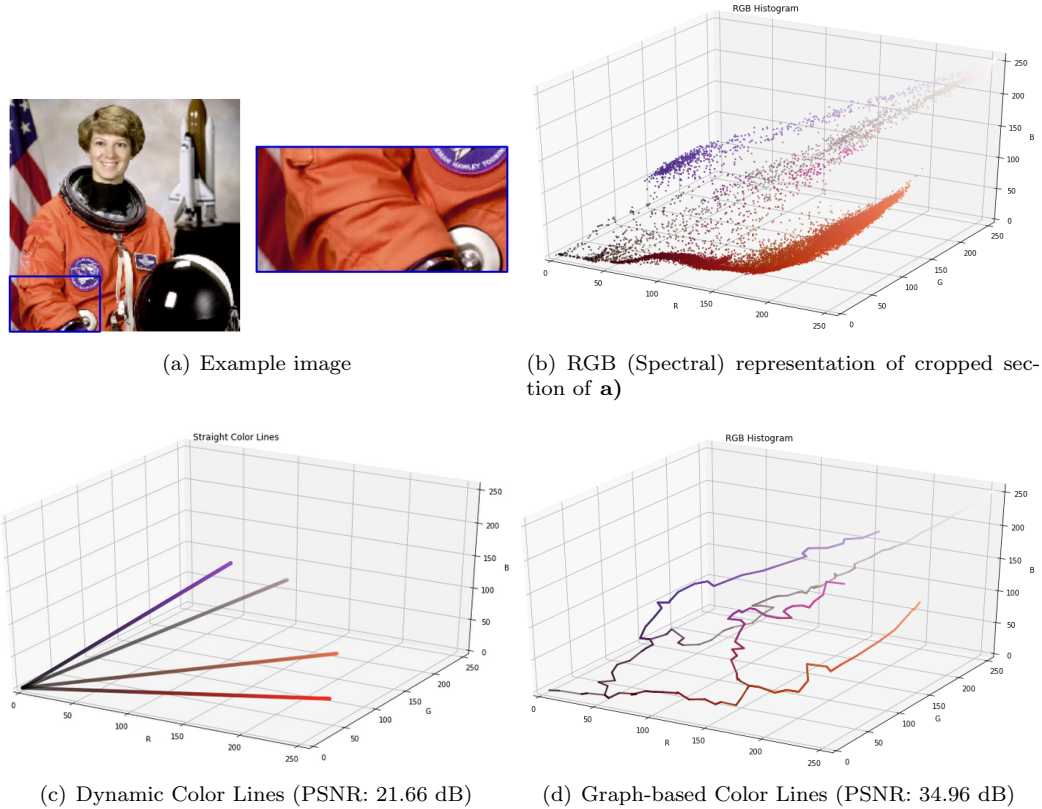


Figure 3.1: Comparison between DCL and GCL. Our model fits spectral representation by piecewise lines.

GCL model relies only on color information. Thus, the simplified representation comprises a set of regions with similar spectrum but “weak” spatial coherence. In order to improve the clustering of similar pixels in more compact regions, we propose to include spatial information in the clustering process. Section 3.4.3 introduces the inclusion of spatial information to the GCL model.

The representation of color images with GCL demonstrated that this approach builds a simplified version that keeps most of the spectral diversity. As an extension to the proposed model, it is presented a simplified representation of higher dimensional images such as multispectral images (MSI) and hyperspectral images (HSI). The representation of MSI and HSI can be useful for tasks such as material recognition and hyperspectral unmixing. Proposed extension of GCL for MSI and HSI is presented in Section 3.5.2.

3.3 State of the art

In this section, we present some relevant concepts related to the representation of color in images (Section 3.3.1 such as color spaces, modeling and descriptors). Then, we report the selected metrics to evaluate our new GCL model from three complementary points of view: evaluation of the whole image, characterization of obtained regions and evaluation of simplified representation as superpixel method.

3.3.1 Color representation

Color is an attribute that allows one to identify one or more objects in images [93]. On the one hand, many efforts have been made to find an adequate color representation via linear and non-linear *color spaces*, as presented by [94]. On the other hand, *Color modeling* plays a different role to model the color diversity as an information vector in a given space.

Color spaces

The RGB (Red, Green, and Blue) color space represents color as the combination of the three primary colors, as presented in Fig. 3.2 (1). In general, cameras used in artificial vision systems acquire images in RGB according to the reflectance of the primary colors. From acquired images, represented as the linear combination of the three channels, actions can be performed pixel-wise such as color discretization. For example, in 3.2 (2) is presented the discretization of the RGB cube in 256 bins by channel and in 3.2 (3) is presented the discretization in 4 bins by channel. According to the application, it is useful to reduce the complexity. In this case, color diversity was reduced from 256^3 to 4^3 .

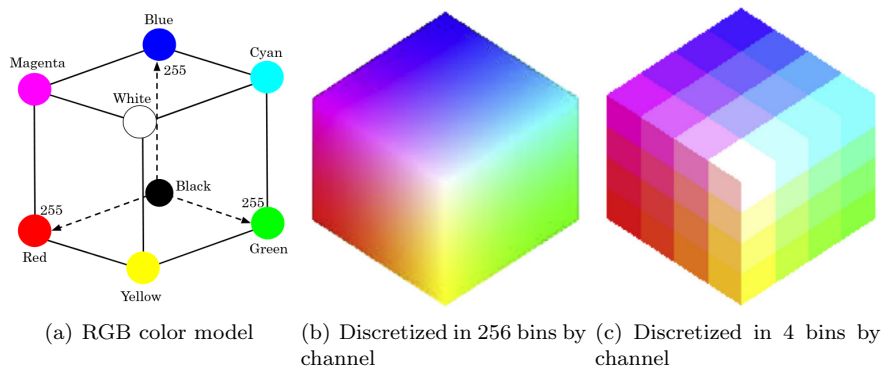


Figure 3.2: RGB color space

Moreover, due to the high incidence of illumination conditions in the image, colors may be strongly distorted using RGB color space. For example, a red object can be seen as a yellow object under certain types and amounts of light under varying light conditions. In order to diversify color representation, it has been developed a significant number of color spaces. As presented by [94], color spaces can be grouped into three categories according to image processing applications:

1. Device-oriented: Widely used in applications that require the color to be consistent with hardware tools such as color TV, storage (CD-ROM), and video systems. Some examples are RGB, YIQ, YCC, Lab and, LUV.
2. User-oriented: Describe the color as the human perceives it. Some examples are HSI (Hue, Saturation, Intensity), HSV (Hue, Saturation, Value), and HLS (Hue, Lightness, Saturation).
3. Device-independent: Created for color instruments such as colorimeters and spectrometers. This category is standardized by the CIE (Commision Internationale d'Eclairage, in french). Some color spaces are: XYZ, CIELAB and CIELUV.

Color modeling

Color is widely used as a relevant feature in image processing, and it is probably the most dominant element to describe a (color) image. It has been observed that computing features from color, such as color histograms, are straightforward. However, processing histograms and performing object recognition may have a high computational cost [95].

Additionally, as stated by [96], color vectors have in general a high dimensionality. The above, together with the loss of spatial information working solely with color, has motivated the development of multiple techniques to model images such as: color lines model [30], representation of color and texture features with Gaussian Mixture Models [97], region-based level segmentation [98], representation of images using body color sets [99], soft-segmentation using color unmixing [100], lattice structures on color spaces [101], and more recently, deep features [102, 103]

Color segmentation

In image processing, segmentation is a well-known operation to partition an image into a set of connected regions. According to the type of application, it may be relevant to group pixels using multiple criteria such as spectral information, range, magnetic resonance, and so on [104]. There are hundreds of segmentation techniques, depending on the application and the type of images. Below are described some of the approaches to segment color images.

Color information

In [105] are summarized and grouped into three categories the different segmentation techniques based on color information: pixel-based, edge-based and region-based schemes. Additionally, some of the most used color spaces for image representation are described. It highlighted the importance of color space selection to reduce the correlation between channels and consequently improve the segmentation. Some of the most popular approaches to image segmentation are described below:

- Based on grayscale images, such as *i)* Histogram thresholding: It is assumed that each object has a different intensity that can be identified as a peak in the histogram; *ii)* Edge detection: Based on big changes in intensity to identify edges of the objects. The main drawback of those techniques is that they may not work well in regions with gradual changes of color because they are strongly dependent on the gradient.
- Region-based approach: They seek to grow and combine regions, as presented by [106, 107, 108]. Some of the advantages are the low incidence of noise and separation in homogeneous

regions. However, it is highly dependent on the initial markers, which can cause errors when growing the regions if the initial markers were not defined correctly.

- Feature space clustering: Each region of the image forms a cluster in feature space. It is straightforward (and simple) to implement. One of its disadvantages is the absence of a method to define the number of clusters in the image. In order to cluster pixels in feature space, it can be implemented using a machine learning model [109].
- Fuzzy approaches: Based on logical rules such as IF and THEN. Its main drawback is that determining the identification of the rules for identifying an object is an extensive task.

Other descriptors

The segmentation of images can be improved by including additional features to color information. Some approaches available in the literature are described below.

Texture information The identification of objects by the human eye is linked to the identification of multiple features in objects such as color, contours, and textures [110]. This motivated the scientific community to the development of techniques that integrate the information color with the texture for the segmentation of the objects. Fig. 3.3(a) shows the evolution of the number of published papers related to color and texture segmentation between 1984 and 2008. For the range between 2000 and 2018, the number of works was measured using Google Scholar. Both graphs show how the segmentation of images using color and textural features is an active research topic.

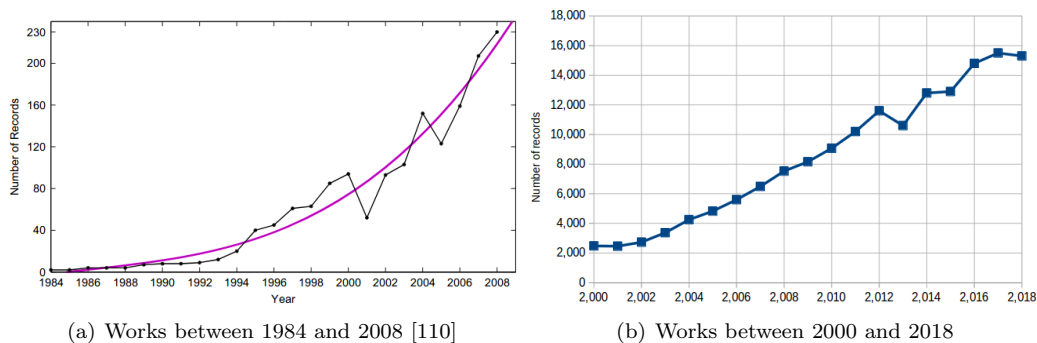


Figure 3.3: Timeline with the number of works on color between 1984 and 2018.

According to [110], techniques of segmentation using texture and color information can be grouped in:

1. Implicit color-texture features integration: In this approach, it is assumed that color and texture features are strongly related so they must be extracted using all image channels. Because the texture characteristics are extracted with multiple filters, it can be useful to reduce the dimensionality of the data with PCA. Techniques grouped in this approach were the first proposed in the area and were extended from grayscale images to color images.

2. Extract color and texture features in succession: This approach surged from the first approach. In this case, proposed techniques do not assume a relation between color and texture features, so that they may be computed in a serial process. Some of the most relevant works in this approach are: Color - texture segmentation from a perceptual point of view [111] and Scale Space Filters (SSF) to partition image histograms [112].
3. Extract color and texture features by channel: The main assumption in this approach is that color and texture are modeled differently when they are analyzed from a statistical view. The techniques can be grouped into two categories: **i)** Region-based integration: Such as split and merge, region growing, and active contours. It assumes spatial coherence between adjacent pixels; **ii)** Statistical and probabilistic-based integration. It assumes that features are locally homogeneous.

Superpixels as simplified image representation

A superpixel is a group of pixels that have some common features such as color and other low-level properties [113]. They allow to build a simplified representation of an image and represent edges better than rectangular patches can do. The use of superpixels is widespread in many image processing applications due to its capability to perceptually represent homogeneous regions of grayscale and color images [114], carry more information than isolated pixels, and provide a compact representation of original images.

Some of the existing superpixels approaches are described below:

- Graph-based algorithms: This approach treats the image as a graph, where nodes are superpixels and edges represent their similarity. In general, the objective is to reduce the cost of cutting the edges of the graph to preserve homogeneous regions. Some of its main drawbacks are related to the lack of control in defining the number of desired superpixels and the compactness of obtained regions.
- Gradient-ascent-based algorithms: This approach iteratively clusters pixels until a convergence criterion is fulfilled. As in the first approach, it does not offer control in the amount and shape of superpixels. Except in Turbopixels [12], where the geometric flows allow to uniformly distribution superpixels over the image.
- Simple Linear iterative clustering (SLIC) [13]: It presents a high performance and low computational cost to compute superpixels. It performs the clustering of pixels according to their similarity using a distance measure in a $5d$ space. Obtained results using this method preserve contour information and are homogeneously distributed over the image.
- Waterpixels [115, 116]: Based on watershed transform [117]. Obtained results using the Berkeley dataset are close to SLIC. This technique has advantages such as uniform distribution over the image, contour preservation, and connectivity between superpixels. Its main drawback is the required preprocessing step because of the placement of the markers in the image. More recently, a Spatial-constrained watershed (SCoW) was proposed by [11].

As a manner of example, in Figure 3.4 is presented a comparison of superpixels calculation with three well known methods: watershed superpixels [11], turbopixels [12] and SLIC [13]. For the interested reader, a detailed survey about superpixel methods can be found in [113]. Also, a most recent review about superpixels applications and datasets to evaluate them is presented by [118].

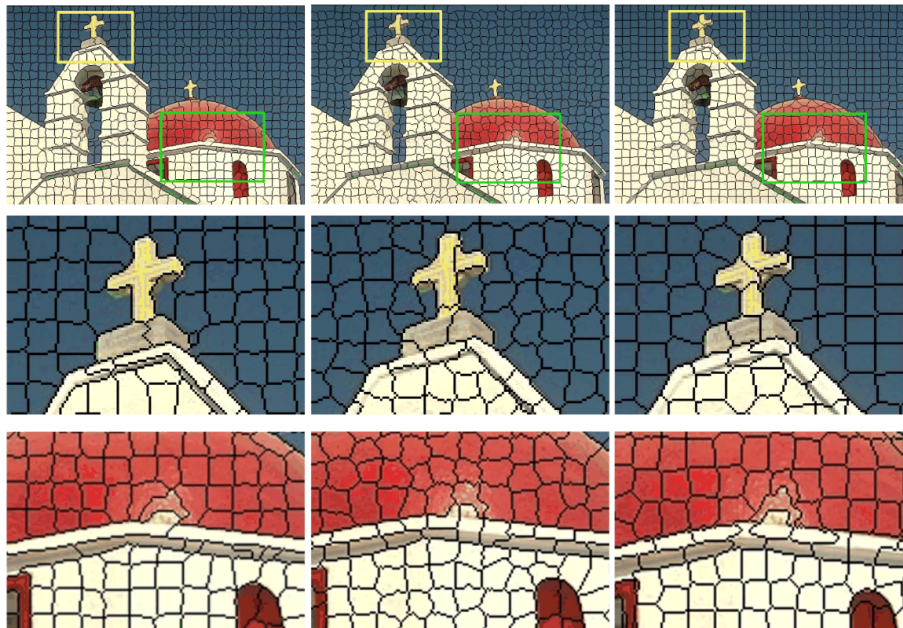


Figure 3.4: Visual comparison of superpixels. Left column: watershed superpixel [11]; center column: Turbopixels [12]; right column: SLIC [13]. Image taken from [11]

3.3.2 Metrics used in this chapter

In this section, selected metrics to evaluate proposed algorithms to represent images in this chapter are presented. They are mainly divided into three groups: 1) Metrics to evaluate the quality of a whole image; 2) Metrics to characterize obtained regions of the image representation.; 3) Metrics to evaluate the simplified representation as a superpixel algorithm.

Metrics to evaluate whole image

As it is presented in Section 3.2, in this chapter is introduced a method to build simplified representation of color images (3.4) and MS/HS images (3.5). The main motivation of the proposed algorithm was to build a reliable representation that preserves the spectral diversity of the original image. Then, in order to compare the quality of the obtained representation (I_p) against the original image (I_i), we selected two metrics that are commonly used as standard fidelity measures [119]: Mean Squared Error (MSE) and Peak to Signal Noise Ratio (PSNR). They are used not only to evaluate the representation of color images but also to representate MSI and HSI.

MSE Mean Squared Error (MSE) is a metric that represents the cumulative squared error between two signals. The MSE is the variance of the estimator in the case of the unbiased estimator. It has the same units of measurement as the square of the quantity being calculated as variance [119].

The calculation of MSE is given by Equation 3.1 [120].

$$MSE(I_i, I_p) = \frac{1}{MN} \sum_{x=1}^M \sum_{y=1}^N |I_i[x, y] - I_p[x, y]|^2 \quad (3.1)$$

Where M and N are the width and height of the image, respectively; I_i is the original image, I_p is the obtained representation. From 3.1, it can be seen that MSE is an absolute metric that will depend on the intensity values of the images. In MSE, the lower the value, the better the performance.

Peak to Signal Noise Ratio Peak to Signal Noise Ratio (PSNR) is a widely used metric to measure the ratio between the maximum possible power value of a signal and the power of distorting noise that affects the quality of its representation. This metric is famous in the field of image compression because it allows us to calculate the ratio between the original image (signal) and the noise is the error included by compression.

The calculation of PSNR is given in Equation 3.2. A *logarithmic* scale is included in the vertical axis due to the high dynamic range of signals.

$$PSNR(I_i, I_p) = 20 \log \left[\frac{\max(I_i)}{MSE(I_i, I_p)} \right] \quad (3.2)$$

Where $\max(I_i)$ is the maximum value of the original image and MSE states for the Minimum Squared Error presented before. In the PSNR metric, the higher the value, the better the performance.

Metrics to characterize regions

In order to evaluate obtained regions of the simplified representation, we selected some metrics that allow us to describe them quantitatively. The following three metrics are computed by region of I_p and do not take into account the original image I_i . They can be interpreted as geometrical properties of the simplified representation.

Fill ratio Fill ratio (FR), also known as “extent”, is computed as the ratio of pixels in the region r_j to the pixels in its bounding box. It is given by Equation 3.3.

$$FR(r_j) = \frac{A(r_j)}{A(BBOX(r_j))} \quad (3.3)$$

Where $A(r_j)$ and $A(BBOX(r_j))$ state for the area of the region r_j and the area of its bounding box. Low values of $FR(r_j)$ indicate very irregular regions and high values mean that regions are well distributed over their bounding boxes.

Solidity Solidity (SO) is computed as the ratio of pixels in the region r_j to the pixels in its convex hull. It is given by Equation 3.4.

$$SO(r_j) = \frac{A(r_j)}{A(CH(r_j))} \quad (3.4)$$

Where $A(r_j)$ and $A(CH(r_j))$ state for the area of the region r_j and the area of its convex hull. Solidity is useful to quantify the amount and size of concavities in an object boundary. Sometimes, solidity can be referred to as *Convexity*.

Metrics for superpixel evaluation

As it is presented in Section 3.4.3, the proposed method in this chapter can be used as a preprocessing step to perform several tasks such as semantic segmentation or object detection in images. In order to measure its performance in over-segmentation tasks, we chose some metrics commonly used in the evaluation of superpixels algorithms [121].

Boundary recall The boundary recall (BR) represents the ratio of superpixel boundaries closer than a given distance d to ground truth edges. It means that this metric is computed by the evaluation of the neighborhood by pixel. In our case, we chose $d = 2$ for all of our experiments, as performed by [122]. The calculation of the boundary recall can be computed through Equation 3.5.

$$BR(I_i, I_p) = \frac{TP(I_i, I_p)}{TP(I_i, I_p) + FN(I_i, I_p)} \quad (3.5)$$

Where TP and FN state for True Positives and False Negatives, respectively. We highlight that they must be calculated in the neighborhood given by the value of d . In this metric, the higher the value, the better the performance.

Undersegmentation The undersegmentation error measures the leak out of superpixel from objects boundaries. There are several implementations to measure undersegmentation errors [12, 122, 121]. Equation 3.6 displays based on Van den Bergh formula for undersegmentation error calculation :

$$UE_{Bergh}(G, S) = \frac{1}{N} \sum_{S_j: S_j \cap G_i \neq \emptyset} |S_j - \operatorname{argmax}_{G_i} |S_j \cap G_i|| \quad (3.6)$$

Where S_j is j -th superpixel, G is segmentation ground truth, and N is the number of ground truth segments. In this metric, the lower the value, the better the performance.

Compactness Compactness is calculated as the ratio of the area of the region r_j to the area of the region of a circle with the same perimeter as r_j , weighted by the ratio of the number of pixels inside the region [123]. The calculation of compactness is given by Equation 3.7.

$$CO(r_j) = \sum_{j=1}^k S_j \frac{A(r_j)}{A(c_j)} \quad (3.7)$$

Where S_j states for the ratio of the area of the region r_j to the area of the whole image, $A(r_j)$ is the area of the region r_j and $A(c_j)$ is the area of a circle with the perimeter of r_j . In this metric, high values represent high regularity in regions shape.

Density Density was initially proposed by [115] in order to measure the ratio of the number of pixels on region contours $|C|$ to the number of pixels in the image N . The calculation of contours density is given by Equation 3.8.

$$DE = \frac{|C|}{N} \quad (3.8)$$

In this metric, a low density indicates a small tortuosity of the regions. In over-segmentation tasks, low DE is desired to prevent too elevated misleading values of boundary recall.

3.4 Representation of color images

The proposed method to build Graph-based Color Lines (GCL) in color images is divided into three stages: the first stage is to build a simplified representation of spectral information; the second stage is a graph modeling of the simplified representation; the third stage is the extraction of GCL. As a descriptive example, Figure 3.1 is used to illustrate intermediate steps in the calculation of GCL.

3.4.1 GCL definition

Simplified color representation

The first stage aims to find a simplified spectral representation of the input image. It follows the same approach proposed by [30] and it is summarized in the workflow of Figure 3.5.

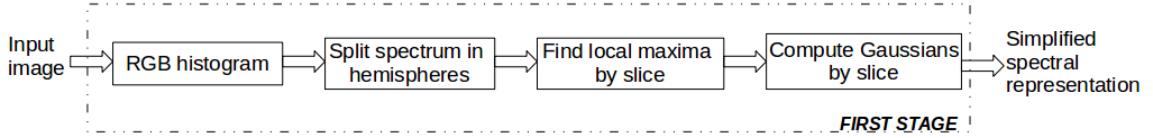


Figure 3.5: Steps for computing simplified spectral representation in GCL model.

Firstly, the RGB histogram is computed, and associated pixels to each color are stored. It is important to highlight that: **i)** Each point of the spectral representation represents a different color in the image. In this work, it is called “color point”; **ii)** A color point may be associated with more than one pixel from the image; **iii)** Color points located further from the origin have a greater luminance. Figure 3.1(b) shows spectral representation from Figure 3.1(a). There are some visually identifiable geometric structures representing related colors in the image. These kinds of structures are common in real-world images as stated in [124]. From the obtained spectral representation, the RGB space is divided into equally separated hemispheres to group the color points according to their luminance seeking to preserve the spectral diversity, as proposed by [30]. The criterion to verify if a color point x_p with color coordinates (r_p, g_p, b_p) is contained in a slice between two consecutive hemispheres with radius r_i and r_j , is given by $r_i \leq \|x_p\| < r_j$. Following the original formulation of the color lines model, a Gaussian mixture model (GMM) is fitted by hemisphere, and the number of components is given by the number of local maxima. It allows us to consider color points associated with colors with high density in the image. Additionally, as proposed by [30], the center of each Gaussian distribution is initialized at each local maximum, removing the instability of random initialization in GMMs. Due to spatial split in hemispheres, GMMs describe

the local behavior of pixels on the slice, and the global spectral diversity of the image is preserved. A Gaussian distribution G_i is given by $G_i = \mathcal{N}(\mu_i, \Sigma_i)$, where μ_i is the mean vector and Σ_i is the covariance matrix of corresponding Gaussian Distribution.

Figure 3.1(a) contains 13.851 color points. After the calculation of the local maxima by slice, the number of Gaussian distributions that describe the spectral representation is $n = 313$ (see Figure 3.6). It is seen that the Gaussian mixture model clustered points of the image with similar colors.

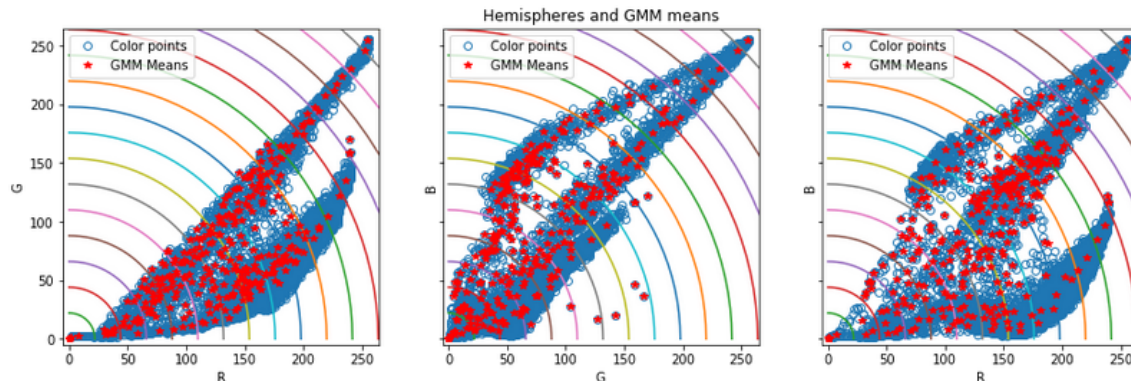


Figure 3.6: Gaussian means of spectral representation of cropped section in Figure 3.1 (a).

Graph modeling of color representation

In the second stage, the input is the simplified representation of the image given by the matrix \mathbf{M} , defined as $\mathbf{M} = [\mu_0, \mu_1, \dots, \mu_n, \mathbf{0}]$, where μ_i is the mean vector of the i -th Gaussian distribution, n is the number of distributions and $\mathbf{0}$ is the origin point vector of the color space found through local maxima of a 3D histogram in the first stage. The expected output is a Minimum Spanning Tree (MST) describing the geometric structures of color points. An approach to generate the color lines from \mathbf{M} is to assume a linear behavior in geometric structures of color points as in [90]. As a result, each color line may be built as an DCL intersecting the origin. Figure 3.7 shows calculated DCL from Figure 3.1. It is seen that obtained color lines do not fit geometric structures of the spectral representation.

The proposed GCL method generalizes [30] to piecewise lines, able to represent more complex geometric structures of spectral representation. In detail, the first step is to calculate the complete graph $\mathcal{G}(\mathbf{M}, \mathbf{E})$ given by: \mathbf{M} each node is a vector mean of the Gaussian distributions and \mathbf{E} as the matrix of Euclidean distances between the vectors in \mathbf{M} . Accordingly, we introduce a graph-based approach by economically linking Gaussian distributions via a Minimum Spanning Tree (MST).

Building graph-based color lines

In the third stage, a graph-based representation from color information is built. We studied different alternatives to link related regions while keeping a physical sense of them. As a solution, we used a graph-based approach to represent each region as a vertex and edges as the link between them, using a MST.

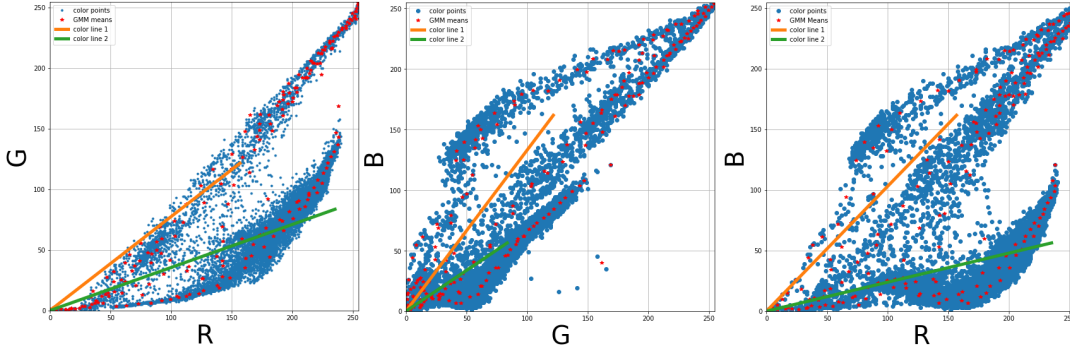


Figure 3.7: First two DCLs of cropped section Figure 3.1 (a).

MST is subgraph of \mathcal{G} that satisfies following conditions: **i)** It spans all the vertices; **ii)** Is acyclic; **iii)** Minimizes the sum of weights of its edges [125]. The use of MST at this point, is motivated for different links between density estimation and the MST from Euclidean distances [126] [127].

As it will be demonstrated later in this section, MST representation preserves the spectral diversity of the simplified representation by the way how *color points* are linked. Moreover, with the proposed GCL model, color lines are defined in the MST, as the path from the origin to each one of the leaves of the tree (nodes of degree one). Each one of these paths is called a *color line* induced by the MST.

The selection of the most important color lines is required to provide a simplified representation of the image. For that purpose, the line importance is ranked according to their length. The length of a GCL is defined as the sum of its edge weights (Euclidean distance between corresponding nodes). A simple algorithm to find the top- K lines employing Dijkstra's algorithm [125] is described in Algorithm 2.

Input : \mathbf{M} (Defined on Section 2.2)

Compute the MST of $\mathcal{G}(\mathbf{M}, \mathbf{E})$;

Add $\mathbf{0}$ to L_1 ;

for $k \leftarrow 1$ **to** K **do**

 Find longest path p from node $\mathbf{0}$ (Dijkstra algorithm);

for each edge $E(i,j)$ in p **do**

if $\mathbf{E}(i,j)$ not visited yet **then**

 Add (j) to L_k ;

 Mark $E(i,j)$ as visited ;

end

end

end

Output: L_1, L_2, \dots, L_K the top- K color lines

Algorithm 2: Compute top- K color lines

Figure 3.8 shows a simple example of the implementation of Algorithm 2 for computing top-3 GCL as follows: **i)** in Figure 3.8(a), the longest path from the origin is composed of $p = [0, 1, 4, 5, 6]$

and so the first color line $L_1 = [0, 1, 4, 5, 6]$; **ii**) After updating weights in Figure 3.8(b), the longest path from the origin is composed of $p = [0, 1, 4, 7, 8]$ and so $L_2 = [7, 8]$; finally, **iii**) After updating weights in Figure 3.8(c), the longest path from the origin is composed of $p = [0, 1, 10, 11]$ and so $L_3 = [6, 7]$; the assumption of assigning common nodes to longest line has some advantages: **i**) Every color point is assigned to a color line; **ii**) Darkest color points (close to the origin) are grouped on first nodes of the longest color line. Assigning common nodes to longest line may have a negative impact in some applications. For example, as it is seen in Figure 3.1 (b), color points close to the origin represent dark colors or shadowed regions. This may mislead to identifying pixels as shadowed sections of the image instead of the actual colors.

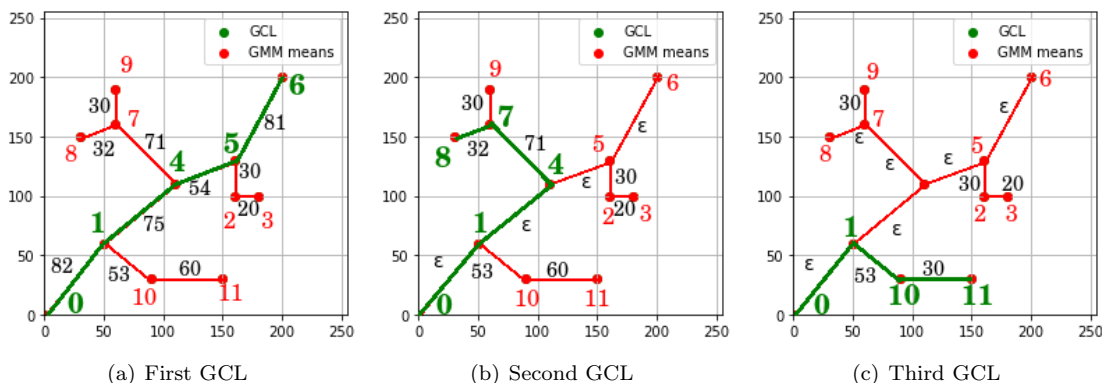


Figure 3.8: Illustrative example of Algorithm 2 to compute top-3 color lines.

As a manner of example of the cropped section of Figure 3.1 (a), the two longest color lines were extracted in Figure 3.9. Each row represents the projection of the spectral representation of the image in the RGB cube (Red vs Green; Green vs Blue; Red vs Blue). In Section 3.4.2 is detailed the procedure to project a color image into the extracted GCL.

There are some relevant findings about this example: 1) Distribution of color points (blue circles) follow some geometrical structures; 2) Computed MST (red line) using color points describes those structures; 3) Extracted color lines adapt to the spectrum geometry. In the next section, experimental results using images and videos are presented.

3.4.2 Experimental Results

This section presents the potential of the proposed GCL model with different applications. In order to evaluate the quality of image representation, the error between obtained image representation and the original image was measured. Different standard fidelity measures such as PSNR and Mean Squared Error (MSE) are used for this purpose. These metrics were introduced in Section 3.3.2.

Projection into GCL

The quality of image simplification using GCL was computed by projecting the original image I_i into obtained color lines. Then, the error between the projected image and the input image is calculated. The procedure for projecting spectral representation is shown in Figure 3.10:

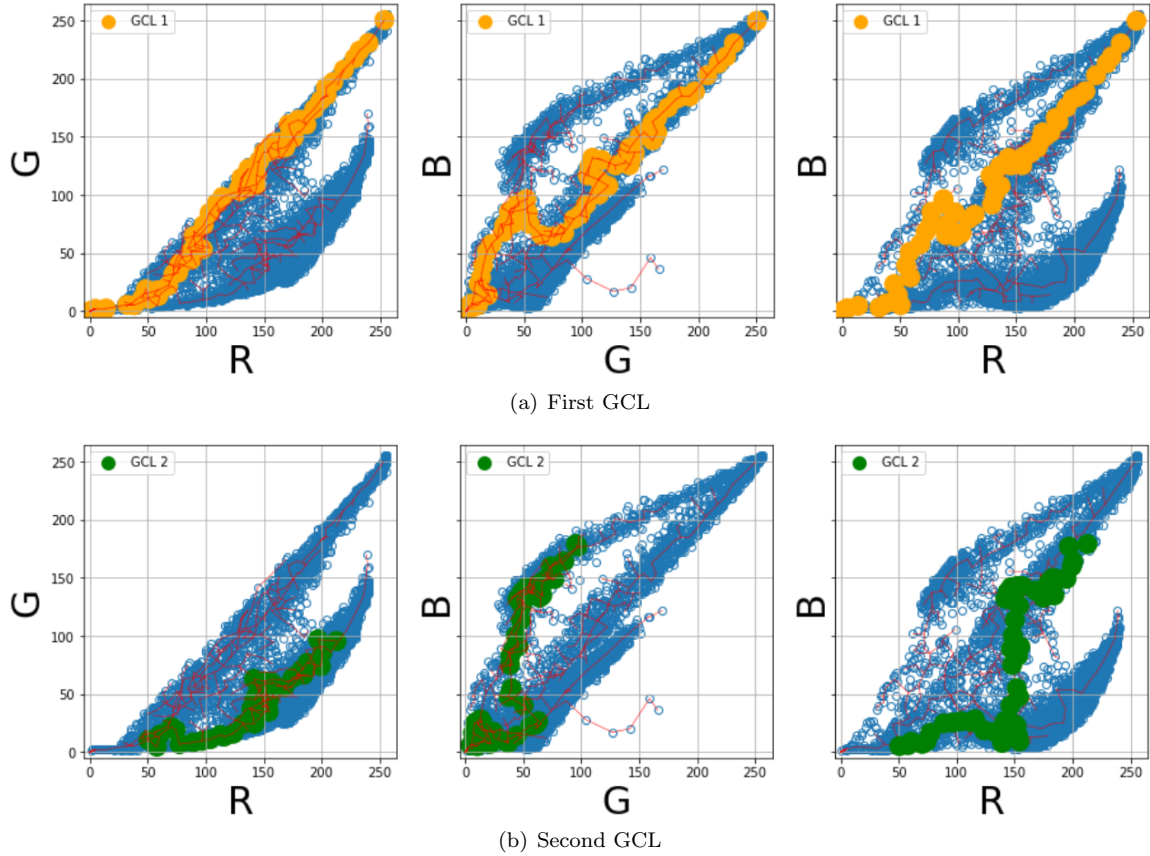


Figure 3.9: Top-2 GCL of cropped section from Figure 3.1 (a), computed in the RGB cube and projected into 2D planes for visualization purposes.

1. *Interpolate GCL*: As presented in Section 3.4.1, each GCL is defined as a sequence of Gaussians of related colors. Piecewise linear interpolations are calculated between means of the Gaussians belonging to the same GCL to add soft color transitions over the line.
2. *Project original image I_i into GCL*: From Figure 3.9, it can be seen that each GCL partially simplifies the color representation. To find the closest GCL for each color point, the Euclidean distance between each color point and every GCL is computed. Then, each color point is replaced by the value of its closest GCL. As a result, color diversity from original representation (I_i) is simplified (I_p) according to the number of GCL.

For quantitative comparisons, Table 3.1 presents the results of PSNR and MSE using DCL and GCL. It is observed that: **i)** PSNR is above 30 dB for images using GCL. The quality of GCL representation is greater than the one obtained using DCL; **ii)** Images 1 and 3 have higher contrast and more different colors. Increasing the number of color lines allows a better fit for more complex

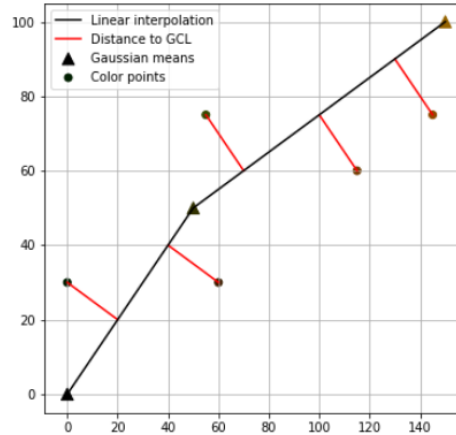


Figure 3.10: Illustrative example of color points projection on a GCL

geometrical structures. An example of fitting a complex structure is shown in Figure 3.9, specially in the second line; **iii)** Images 2 and 4 do not present considerable variations on PSNR and MSE when the number of color lines increases. It occurs because the spectral representation of the image has few geometric structures, so few color lines are required to model the image. Table 3.2 shows percentages of pixels grouped into each GCL with 10 hemispheres. From obtained results, it was found that in some images such as 1, 3, and 4 from Table 3.2, the percentage of pixels grouped in the first four lines is over 85%. In the same table, the line 4 of the image 1 groups 57.86 % of the points. It occurs because most of the color points of the image are orange and are not far from the origin to be clustered in the longest lines.





Id	Image	K	PSNR (dB)		MSE	
			DCL	GCL	DCL	GCL
1		3	20.19	17.68	1864	3328
		4	21.65	34.96	1333	62
		5	21.65	35.14	1333	59
2		3	21.44	33.54	1380	86
		4	21.56	34.10	1361	75
		5	21.57	34.11	1366	75
3		3	27.93	30.56	314	171
		4	28.56	35.96	271	49
		5	28.70	35.99	262	49
4		3	20.29	30.51	1822	173
		4	20.34	32.63	1800	106
		5	20.36	32.65	1794	105

Table 3.1: PSNR and MSE. DCL vs GCL

Id	% of pixels by line				
	L_1	L_2	L_3	L_4	Sum
1	8.21	17.75	1.6	57.86	85.41
2	69.45	1.96	0.01	1.79	73.2
3	76.58	0.19	7.81	3.5	88.07
4	86.71	5.27	0.82	1.03	93.83

Table 3.2: Percentage of pixels in first four GCL using 10 hemispheres with all images.

Figure 3.11 displays the relationship between the number of hemispheres (varying from 1 to 40) against the number of GCL (varying between 1 and 20), of full Astronaut image from Figure 3.1(a). As it is seen on the top of Fig 3.11, a reduced number of GCL and a small number of hemispheres represents the lowest image quality. The former because spectral diversity is oversimplified with few GCL. The latter because the GMMs cluster most dense regions in RGB cube. Nevertheless, increasing both (lines and hemispheres) does not allow indefinite improvement of the image simplification quality. In this example, increasing above 9 the number of lines or above 8 the number of hemispheres does not raise the PSNR value.

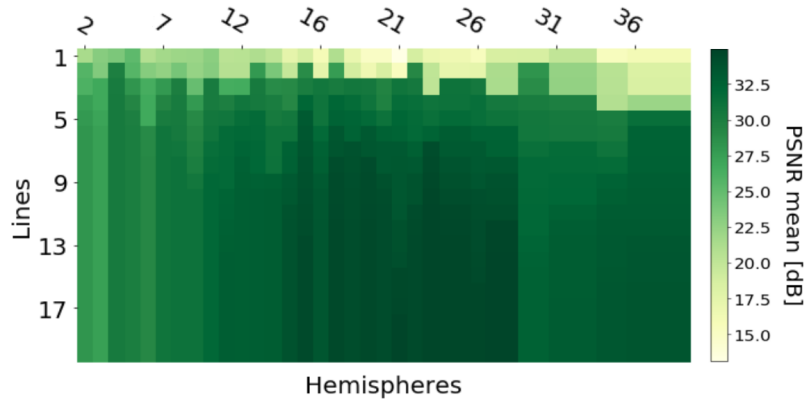


Figure 3.11: PSNR varying the number of GCL and hemispheres from Table 3.1.

To show the *capacity of generalization* [33] of the method, we performed an experiment to measure PSNR to evaluate the quality of reconstruction in a video. The following experiment was carried out: the first image I_0 of the video was simplified using the GCL model. Then, the rest of the images of the video were projected into the already computed GCL and their PSNR was calculated. Note that an excellent model fitting does not guarantee good performance in predicting future frames, because of possible over-fitting on I_0 . The selected videos were taken from [1] (*boxing-fisheye* and *mallard-fly*) and the **top-5 GCLs** on the first frame (I_0) were calculated. Figure 3.12 shows PSNR and MSE over the frames of each video. One can observe that PSNR and MSE were stable in the *boxing-fisheye* video. It occurs because the video corresponds to a sequence of frames without significant changes. In the video *mallard-fly*, the error increases in the last frames. It occurs because there is camera panning and the last frames have nothing in common with I_0 , where GCL were calculated. More examples of GCL for different videos of [1] are available at

<https://people.cmm.minesparis.psl.eu/research/duque/gcl>

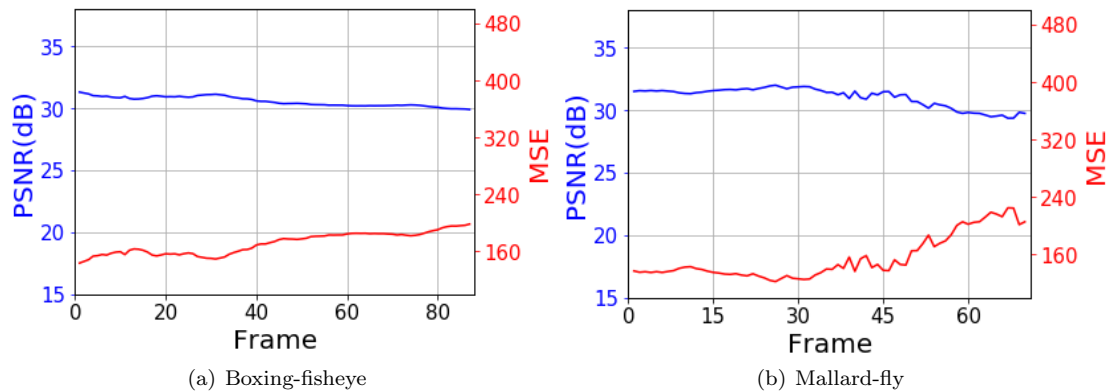


Figure 3.12: PSNR and MSE in two videos from [1]. PSNR (blue) and MSE (red).

Figures 3.13 and 3.14 show the first frame from videos of Figure 3.12. It is seen how the color diversity changes according to the number of GCL in the model. Even though, as is presented in Figure 3.11, adding GCL to the model does not guarantee a gain in PSNR.

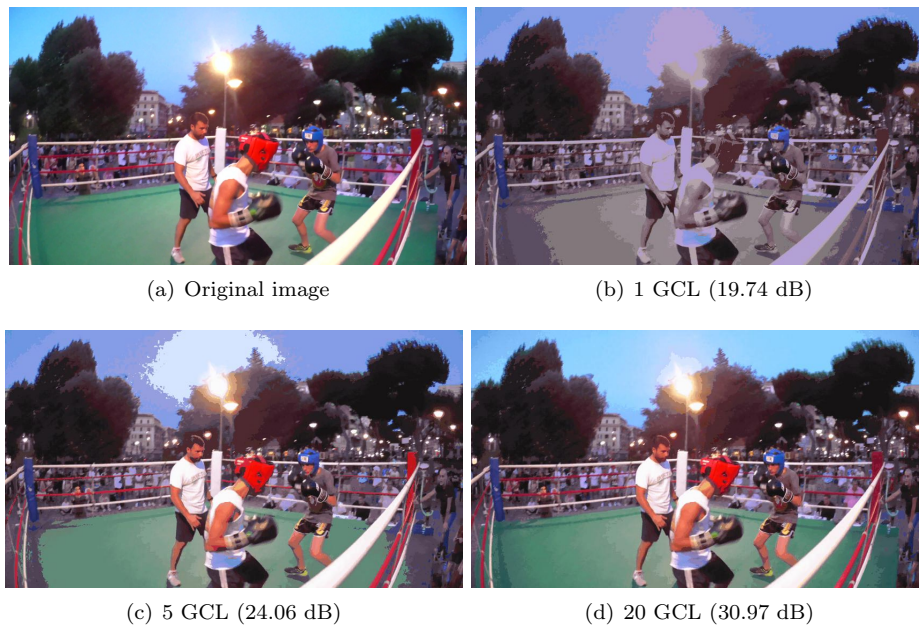


Figure 3.13: Images from boxing-fisheye [1].



Figure 3.14: Images from mallard-fly [1].

3.4.3 Including spatial information to GCL

We studied some alternatives to the methodology proposed in Section 3.4.1 to improve image simplification. As presented in Section 3.4.1, the first stage calculates a simplified representation of the color image using Gaussians. Obtained results demonstrate that using only spectral information, high values of PSNR in image reconstruction are achieved. However, it is also observed that an approach based on color information leads to pixel clustering in noncompact regions. Additionally, it does not preserve information about contours and may group in the same Gaussian distant pixels. In order to improve the contours, the work-flow of Figure 3.5 is modified to Figure 3.15. This process is inspired by the combination of color and spatial information proposed by the well-known SLIC superpixels approach [13].

Including spatial channels

As it is presented in Figure 3.15, the first step is to include spatial information as two additional channels of the image. In the same spirit of some geometrical based superpixels [13, 12], (x, y) coordinates enhance compactness and connectivity of regions. In the case of GCL, it was included a parameter s to weight spatial information in the model, defined as follows: *i)* $s = 0$, only use color information; *ii)* $0 < s < 1$, spatial channels have lower weight than color channels; *iii)* $s > 1$, spatial channels weight more than color channels.

If the value of s is greater than 0, two channels are added to the color image I . It increases the dimensions of the image from $[h, w, 3]$ to $[h, w, 5]$. It evidences a considerable augmentation in data quantity. This variation modifies histogram calculation and considerably increases the

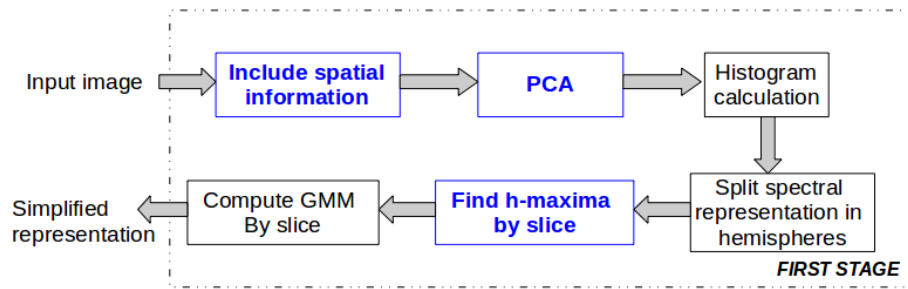


Figure 3.15: Modified diagram including spatial information to the simplified representation presented in Figure 3.5.

computational time. As an additional step, a Principal Component Analysis (PCA) is included to deal with those issues.

Principal Component Analysis (PCA)

Dimensionality reduction using PCA is an approach in image processing [128, 129]. A PCA was included to reduce the dimensionality of the image I and select the first three components. In Figure 3.16 is presented an example of an RGB image projected over its Principal Components. One may note some differences in projected space compared against RGB cube: *i*) Negative values *ii*) The origin corresponds to the mean of the data and not necessarily to black color; *iii*) Points with the highest norm do not represent colors with the highest luminance but the most different pixels compared to the mean.

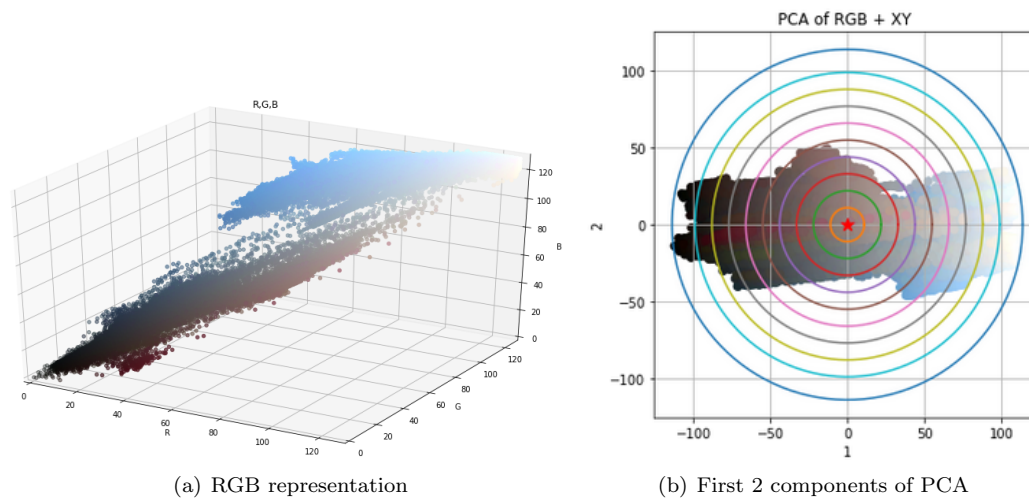


Figure 3.16: Dimensionality reduction with PCA.

As presented in Section 3.4.1, some of the assumptions of building original GCL to model color

diversity in images are: *i*) CL intersect at the origin (black); *ii*) Similar colors are uniquely located in the RGB cube. Due to the differences obtained as a result of projecting PCA, the main idea of building CL as a representation of color diversity in the image is strongly affected, and they cannot be computed as before using a graph-based approach. To build the CL with the same approach leads to a graph-representation where each line starts at the middle of the image because of the computed components obtained in the PCA. So, in this case, the simplified representation of the image is given by the GMM that models spectral diversity with spatial information.

Compute *h*-maxima The output of the first stage is a simplified representation of the color image (Section 3.4.1) using Gaussians to group *related pixels*. When only spectral information was used, the Gaussians were initialized in the maxima of color histograms. However, when spatial information is included in the representation, an augmentation of the number of maxima in the representation occurs. It occurs mainly by two reasons: first, because the spatial information is completely unrelated to spectral information; second, because PCA transformation finds the direction of uncorrelated variables dependent on the (x, y) coordinates. The increase of the number maxima impact our proposal because it is used to define the number of Gaussians of the model.

In order to restrict the number of Gaussians and select the most relevant maxima, it was implemented *h*-maxima of the density [14]. The main idea behind this transformation is to discard maxima with lower contrast than a threshold value h . For example, as presented in Figure 3.17, if $h_1 < h$, then the maximum is preserved. If not, it is discarded. The choice of using *h*-maxima transform includes a parameter h to establish the minimum contrast of the maximum to initialize a Gaussian there.

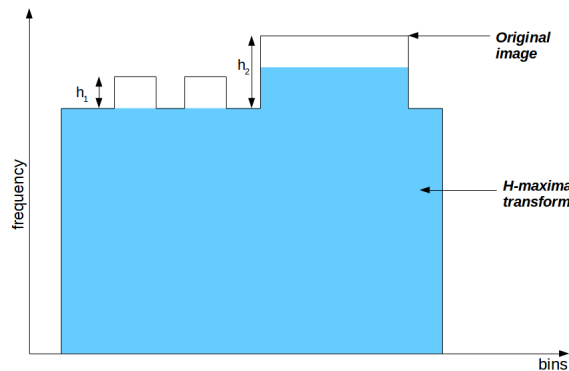


Figure 3.17: *h*-maxima transformation [14].

Experimental Results

Some experiments were performed to evaluate image simplification, including spatial information with RGB images from [2]. The primary motivation to include (x, y) was to add information about the position of pixels in the image to get compact regions. On the one hand, it is expected to decrease the quality of image reconstruction because distant pixels with similar spectral information will not be grouped into the same Gaussian. On the other hand, it is expected to obtain more compact pixel regions with better contour preservation.

Moreover, to measure the incidence of the spatial information in the shape of the simplified representation, two additional metrics were included: extent (fill ratio) and solidity. These metrics are introduced in Section 3.3.2.

As a demonstrative example of the incidence of spatial information, Figure 3.18 shows the simplified representation of a color image varying the value of s between: $s = 0$, $s = 0.5$ and $s = 1$, from left to right. As is expected, region shapes change with the value of s . The higher the value of s , the more compact the regions. For example, in Figure 3.18 (a), the shape of regions of the ground preserves contours and group pixels with similar spectral and spatial information. However, the quality of reconstruction decreases according to PSNR values. Our proposed representation permits to build a reliable simplified representation with high values of PSNR when no spatial information is included because it group related pixels even if they are not closer in the image. Additionally, when spatial information is included, we can observe that most of the contours are preserved and the pixels are grouped in the same Gaussians not only with their color similarity but also by their spatial proximity.

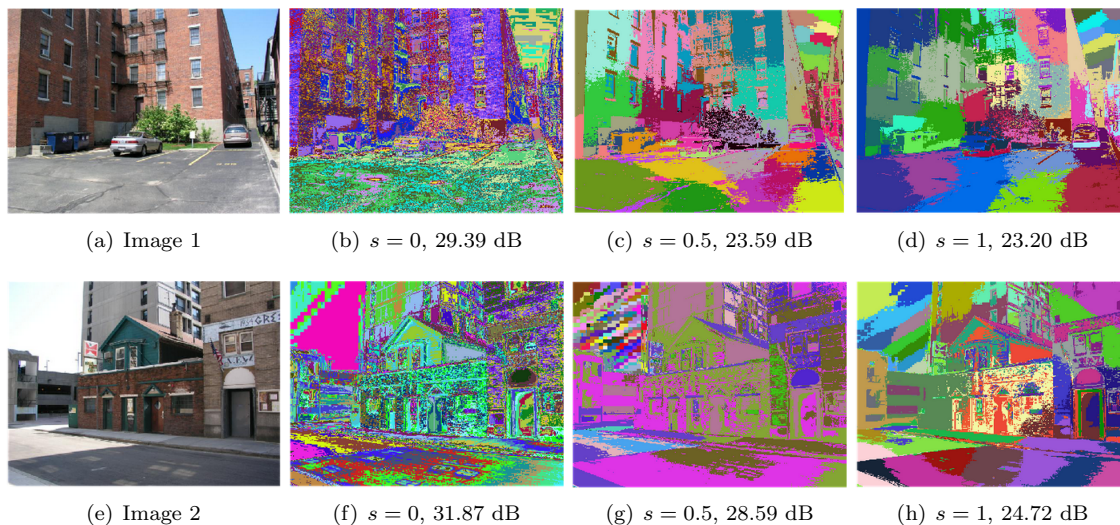


Figure 3.18: Example of image representation varying the weight of spatial information s . It is observed how the shape of regions changes according to s value. The use of $s > 1$ leads to regions grouping pixels based on their spatial proximity and less preservation of contours.

In order to evaluate the incidence of spatial weight in several cases, we performed some experiments using images from the BSD dataset [3]. The variation of s changes the number of maxima in the representation. To perform a fair comparison between the simplified representations, the value of h was adjusted by experiment to have around 150 Gaussians at each time. The above, seeking to obtain a similar number of regions in each experiment, can get a comparable complexity of different models to get a fair comparison. In this case, we reported: PSNR between the original image and the simplified representation, using the projection presented in Section 3.4.2, extent and solidity. In Table 3.3 we report obtained results for training set from BSD500 [3]. In each cell, we report mean values and standard deviations.

From Table 3.3, it is seen that the characteristics of the regions change according to the weight

Spatial weight	PSNR	Extent	Solidity
0.0	28.469 \pm 4.1752	0.0251 \pm 0.0235	0.0326 \pm 0.0282
0.2	27.037 \pm 3.4744	0.0762 \pm 0.0488	0.1071 \pm 0.0644
0.4	26.088 \pm 2.8916	0.1339 \pm 0.0541	0.1923 \pm 0.0736
0.6	25.056 \pm 2.5853	0.1750 \pm 0.0563	0.2514 \pm 0.0769
0.8	24.092 \pm 2.2104	0.2021 \pm 0.0621	0.2923 \pm 0.0841
1.0	22.997 \pm 1.9630	0.2277 \pm 0.0673	0.3310 \pm 0.0929

Table 3.3: Metrics of obtained Gaussians in BSD500 dataset varying spatial weight of our proposed image representation and SLIC algorithm [13].

Method	R	P	DE	CO	US
0.0	0.9995 \pm 0.0017	0.1293 \pm 0.0443	0.2812 \pm 0.1334	0.0162 \pm 0.0032	0.9101 \pm 0.0748
0.2	0.9977 \pm 0.0047	0.1418 \pm 0.0479	0.2257 \pm 0.1144	0.0269 \pm 0.0105	0.7906 \pm 0.1245
0.4	0.9933 \pm 0.0161	0.1502 \pm 0.0505	0.1900 \pm 0.0922	0.0369 \pm 0.0177	0.7077 \pm 0.1464
0.6	0.9891 \pm 0.0193	0.1546 \pm 0.0530	0.1782 \pm 0.0825	0.0459 \pm 0.0239	0.6606 \pm 0.1483
0.8	0.9848 \pm 0.0252	0.1579 \pm 0.0547	0.1735 \pm 0.0802	0.0543 \pm 0.0305	0.6232 \pm 0.1503
1.0	0.9742 \pm 0.0757	0.1619 \pm 0.0556	0.1720 \pm 0.0858	0.0683 \pm 0.0580	0.5830 \pm 0.1542
SLIC	0.7837 \pm 0.0766	0.1861 \pm 0.0657	0.0737 \pm 0.0342	0.5336 \pm 0.1266	0.1818 \pm 0.0157

Table 3.4: Results in BSD500 dataset varying spatial weight of our proposed image representation and SLIC algorithm [13]. R: Recall, P: Precision, DE: Density, CO: Compactness and US: Undersegmentation error

of spatial information. Reconstruction quality, measured with PSNR, decreases with higher values of spatial weight. It is an expected performance because a pure spectral approach clusters pixels in the same Gaussian even if they are not spatially connected. Extent and solidity values increase with higher spatial weights because obtained Gaussians group close pixels. This is an expected result because it demonstrates that more reliable regions are obtained to perform feature extraction using this type of image simplification.

Oversegmentation An application of the proposed algorithm to represent color images is an over-segmentation task. In this section, we present experimental results in BSD500 dataset [3] where we evaluated the incidence of the spatial weight in contours detection. We selected a train set composed of 200 images and calculated image simplification, varying the spatial weight s between 0 and 2 with a step of 0.2. If $s = 0$, regions do not include spatial information; if $s = 1$, spatial information weights the same than spectral information. If $s > 1$, regions group pixels based mainly on their spatial proximity with less preservation of the contours. As presented before, we adjusted the value of h to have around 150 Gaussians by image.

In Table 3.4 we present obtained results with our proposal and compared it against results obtained with SLIC [13] method. In each cell, we report mean values and standard deviation.

3.4.4 Discussion about color images

Section 3.4.1 of this chapter presents the proposed method to model spectral diversity of color images in the spirit of Color Lines [30]. The resulting GCL model performs better than DCL [90]

in most of the explored examples, which is natural due to the higher complexity of the proposed model (piecewise linear function instead of straight lines). However, experiments on videos show that the GCL model computed on a single frame does not over-fit and correctly describes the whole sequence within the same scene.

The Algorithm 2 proposed to build color lines from MST was illustrated with a simple example in Figure 3.1. One of the main drawbacks of assigning common nodes to the longest line is to identify some pixels as shadowed regions instead of the actual colors. One alternative to solve this issue is to perform a preprocessing stage of the image in another color space such as HSV or HSV before computing GCL model.

From obtained results of the GCL model based only on spectral information, it was proposed a variation of image simplification to include spatial coordinates in Section 3.4.3. This inclusion strongly increased the volume of data and the calculation of local maxima by slice to initialize Gaussian. We proposed to use the h-maxima transformation [14] that permits to select of only the most relevant maxima. It allows the use of h parameter as a criterion to the number of Gaussians in the representation. The value of h controls the number of Gaussians in the representation and the computational cost of the method. Additional studies about the computational cost of h vs performance can be helpful.

The inclusion of spatial information improved the compactness of the regions and preservation of the contours in the model. It includes a parameter s to indicate the relative weight of spatial information compared to spectral channels. As it is presented in Figure 3.18, the value of s affects the obtained image representation. The value of s is more intuitive to set because it weights the importance of (x, y) coordinates in the model. However, to tune this value depends on the application. For example, for image compression tasks, low values of s lead to simplified representation where most of the spectral information preserved and the highest PSNR are obtained. However, for other applications such as oversegmentation or object detection, it is desired to have spatially connected regions and a higher value of s must be assigned.

As we presented before in this chapter, the main motivation of GCL was to build a simplified representation that preserves spectral information of images. We evaluated our proposal with some of the most used metrics for superpixels algorithms and compared them against SLIC. From Table 3.4, it is observed that a pure spectral model (with $s = 0$) does not take into account spatial proximity between pixels and does not preserve contours, as expected. However, when the value of s increases, all metrics are improved. This is the desired effect of including spatial information into the clustering of pixels on Gaussians.

The proposed method to represent color images can be applied to higher-dimensional data such as multispectral (MSI) and hyperspectral images (HSI). In Section 3.5, we present the extension of GCL to this kind of data. Also, as it was introduced for color images, spatial information was included to improve image representation quality.

3.5 Extension to MSI and HSI

This section extends the technique to represent color images using GCL to high dimensional images such as MSI or HSI. The proposed methodology is described following two approaches: *i*) Using only spectral information; *ii*) Using spectral and spatial information. With both simplifications, several alternatives to cluster pixels in regions were evaluated. The motivation behind the contribution is to simplify higher-dimensional images in order to work in several applications such as *material recognition* or *image segmentation*.

The extension of GCL presented is organized as follows: in Section 3.5.1, the proposed methodology to simplify HSI is described. In Section 3.5.2, performed experiments with images from University of the Negev [7] and Powders dataset [8], are described. Finally, Section 3.5.3 introduces some improvements to image representation.

3.5.1 GCL on HSI/MSI

The proposed methodology to build a simplified representation of MSI / HSI is mainly divided into two stages: 1) Reduce image complexity; 2) Group similar pixels. The methodology was evaluated using the powders dataset [8] in order to measure the quality of image reconstruction. As it is demonstrated for color images, the main interest of the proposed image representation is to preserve spectral diversity, even if the regions where the powders are located in the image are small. As a descriptive example, Figure 3.19 shows the simplification of one HSI with powders. It can be seen how the regions of the image with powders in (a) are modeled by one or more Gaussians in (c).

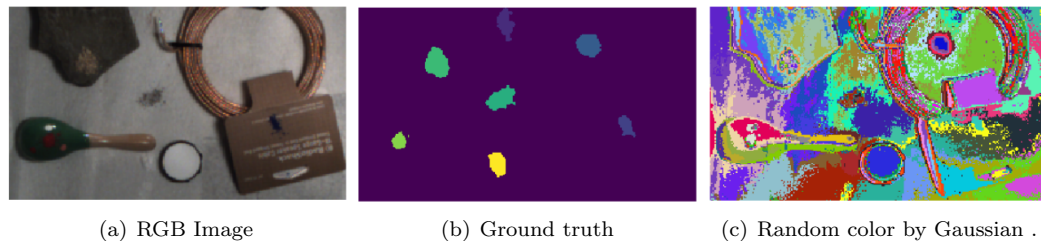


Figure 3.19: Simplified representation using a hyperspectral image from powders dataset with 79 Gaussians. PSNR: 32.94 dB.

Reduce image complexity

A high dimensional image can be represented as a cube I with dimension h, w, n , where h is image height, w is image width, and n is the number of channels acquired. Several works have demonstrated that a elevated volume of data and its sparsity in HS images are a common issue [130, 131]. We proposed to reduce image complexity by using two steps as follows:

- *Dimensionality reduction*: A PCA is computed to identify the most relevant and orthogonal axes of spectral information in I . For the experiments presented in Section 3.5.2, the first three principal components were selected because they accounted for more than 95 % of the variance from input data.
- *Histogram*: They are calculated using the first three components obtained using PCA.

Figure 3.20, (a) shows the projection of two of the three components computed using PCA in a hyperspectral image and (b) shows the 2D projection of the histogram of the three components. It can be noted how the density of pixels in Figure 3.20 (b) is greater close to the origin of the PCA projection of Figure 3.20 (a). It occurs due to the dimensionality reduction with PCA that concentrates most of the information around the mean.

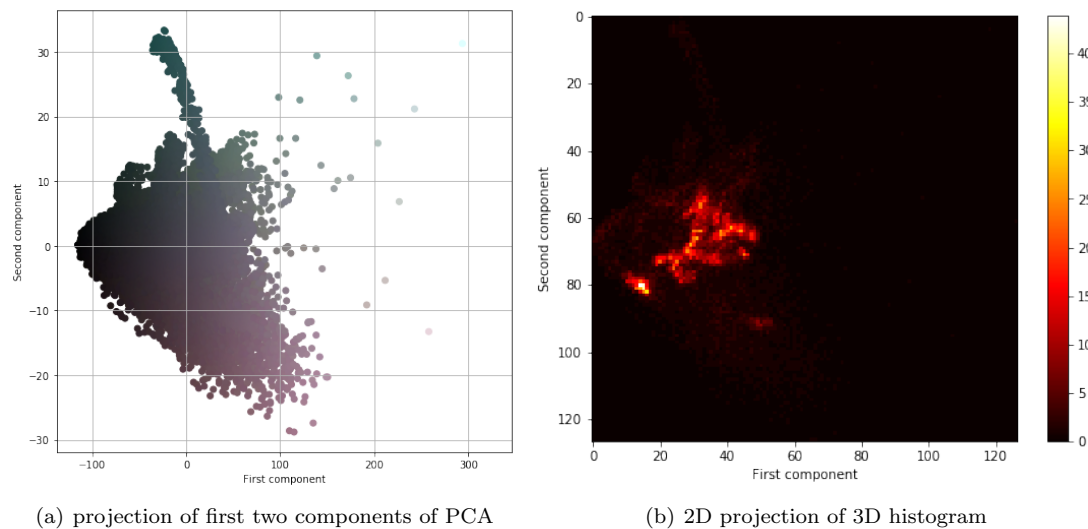


Figure 3.20: PCA and projection of 3D-histogram of hyperspectral image

Group similar pixels

The second stage takes as input the computed histogram from Section 3.5.1. It should be noted that the original hyperspectral image I of dimensions $[h, w, n]$ has already been simplified to a p dimensional array H , where $p = 3$ is the number of components selected from PCA.

In order to group related pixels from the original image using spectral information, a GMM by slice inspired by GCL was fitted. Then, the number of Gaussians by GMM was obtained using h -maxima of the density (as before with color images with spectral information). In all the experiments, we used $h = 3$ that demonstrated experimentally to be a good compromise between computational cost and image quality. It must be noted that each group of pixels clustered in the same Gaussian represents a region of the image with similar spectral information. The following sections describe three approaches varying the initialization parameters and the set of pixels selected to fit a GMM.

Approach 1: Random initialization of GMM In the first approach, we selected the entire set of pixels from Figure 3.20 to fit a GMM. Due to the selected implementation of GMM [132], a k -means algorithm is performed if the initial centers of the Gaussians are not specified.

Approach 2: GMM with h -maxima initialization In the second approach, we selected the entire set of points from Figure 3.20 to fit a GMM. In this case, unlike the first approach, the centers of GMM were initialized with h -maxima of the histogram.

Approach 3: GMM on slices As the third approach, the representation of Figure 3.20 was split into equally separated hemispheres. This separation was performed in the spirit of CL. Then, according to the value of pixel norm, they are grouped in slices, and a single GMM is fitted by each

one. As demonstrated in CL, this separation preserves the spectral diversity of the image. As a manner of example, Figure 3.21 illustrates the clustering of color points using this approach.

In order to compare the performance of proposed approaches, the number of Gaussians of the first approach is defined as the number of maxima selected by the h -maxima approach. This assumption allows us to have the same number of Gaussians in all representations.

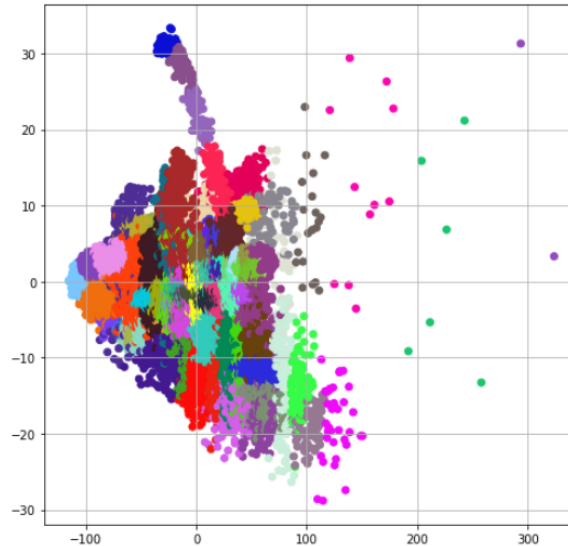


Figure 3.21: Clustering of color points using the third approach introduced in Section 3.5.1.

3.5.2 Experimental results

This section presents obtained results of image simplification in MSI and HSI using the proposed method. As a measure of the quality of the obtained representation, PSNR between the input image and the simplified image is considered. Some experiments using images of powders acquired in a laboratory with controlled light conditions [8] were performed. The size of each image is $h = 160$, $w = 280$ and $n = 961$. Due to the high number of channels (HSI), the authors of the dataset also provided a reduced version composed of images with 12 channels (MSI). Performed experiments in this section were executed with those images.

Table 3.5 presents PSNR values of six HSI. First, the quality of the reconstruction over the whole image (rows with label a) was measured. Then, the quality only in the region where powders are located (rows with label b) was computed. The number of regions is given according to the value of h of h -maxima. One can observe that Approach 3 better represents these regions in four of the six images, even if the quality for the whole image is lower.

In performed experiments, it is seen that when the Gaussians are constrained by slice (10 slices in all cases), the model covers the spectrum better and preserves spectral diversity. This is important in some applications, such as the one presented in [8]. In that application, the relevant information is the presence, or not, of some powders. These powders can be present in small quantities, and

	Id	# Gaussians	Ap. 1	Ap. 2	Ap. 3
a	1	48	22.17	22.07	21.83
	2	67	30.07	30.12	29.99
	3	64	27.47	26.75	26.89
	4	67	31.48	31.14	30.89
	5	61	21.28	21.24	19.31
	6	51	26.7	24.55	26.82
b	1	48	31.72	32.35	32.94
	2	67	44.01	44.76	44.89
	3	64	36.60	38.07	37.41
	4	67	47.62	47.94	48.91
	5	61	25.44	25.42	25.35
	6	51	32.22	32.69	33.17

Table 3.5: Simplification of six HSI using proposed approaches. Error measured in: whole image (*a*) and regions with powders (*b*). It is observed how the image simplified keeps spectral diversity of the original image. The quality of reconstruction is evaluated by means of PSNR.

then the classical image simplification may discard them.

3.5.3 Improvements to GCL in HSI/MSI

Including spatial channels (x, y)

As presented in Section 3.4.3 for color images, the inclusion of spatial information increases region compactness and preserves information about pixel location. In a similar way as proposed in color images, (x, y) coordinates were included as two additional channels in MSI and HSI.

The proposed methodology in Section 3.5.1 uses only spectral information of images to calculate simplified representation. To include spatial information in hyperspectral images, the technique is similar as presented in Figure 3.15 for color images.

As an illustrative example, Figure 3.22 presents the simplification of a HSI from [7]. In this case, image simplifications have about 180 regions. After the inclusion of spatial information, it can be observed in Figure 3.22(c) that: *i*) Pixels are grouped in compact regions (can be useful for feature extraction); *ii*) Value of PSNR decreases because regions are constrained to be spatially connected. This last finding was also evidenced with color images in Section 3.4.2.

According to the selected information to compute image simplification, it is noted by: **S**: Only spectral information; **S-XY**: Spectral and spatial information.

From multiple GMM to single GMM

In Section 3.5.1 is presented a methodology to simplify MSI/HSI in the spirit of Color Lines. One of the characteristics of this model is the constraint of clustering pixels only within Gaussians of the same slice. However, this constraint may reduce the quality of image reconstruction in certain scenarios. For example, pixels close to slice borders should be classified in Gaussians of neighboring slices instead of being clustered by a Gaussian of the same slice in some cases. Figure 3.23 shows an indicative example of the previously mentioned issue.

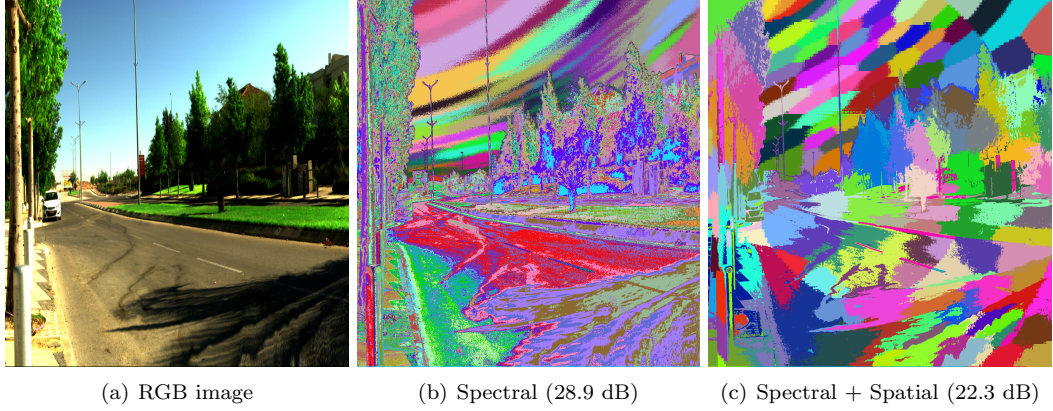


Figure 3.22: Simplified representation of hyperspectral image

As described earlier in Section 3.4.1, the spectral space is divided into equally separated hemispheres. Then, a GMM is calculated by slice to preserve the spectral diversity of the image. It implies that calculating the posterior probability of all the Gaussians in the space cannot be directly completed. To solve it, we proposed to group all Gaussians of the space into one single GMM. By definition, a GMM is a linear combination of Gaussians (see Eq. 3.9), where k is the number of Gaussians in the model, w_i is the weight, Σ_i is the covariance matrix and μ_i is the center of Gaussian i . From already computed Gaussians by slice, it can be directly calculated: **i)** Matrix μ_i with Gaussians centers. It is a $2D$ Matrix, with dimensions $[m, p]$, where m is the number of Gaussians and p is the number of principal components selected; **ii)** Matrix Σ_i with Covariances matrix of each Gaussian. It is a set of m matrices with dimensions $[p, p]$; **iii)** Vector w_i with Gaussians weights. It has length m and a value of $1/m$ for each element.

$$G(\Phi|\mathbf{x}) = \sum_{i=1}^k w_i \mathcal{N}(\mu_i, \Sigma_i) \quad (3.9)$$

All the Gaussians of the space are grouped into a single GMM given by Eq. 3.10. Some of the advantages of grouping all Gaussians in G_g are: **i)** Directly computation of posterior probability; **ii)** Sum of all probabilities is normalized; **iii)** Reduce computational cost because class prediction is performed in one single step.

$$G_g(\Phi|\mathbf{x}) = \frac{1}{m} \sum_{i=1}^m \mathcal{N}(\mu_i, \Sigma_i) \quad (3.10)$$

After the calculation of G_g , pixel clustering is performed. As Figure 3.23 (b) shows, most of the significant changes should occur close to the borders between hemispheres. As a result of this modification, it is expected an increase of PSNR value in image reconstruction.

According to the selected information for clustering pixels, it is noted: **C**: Constrained to Gaussians of the same slice; **NO-C**: Not-constrained to Gaussians of the same slice.

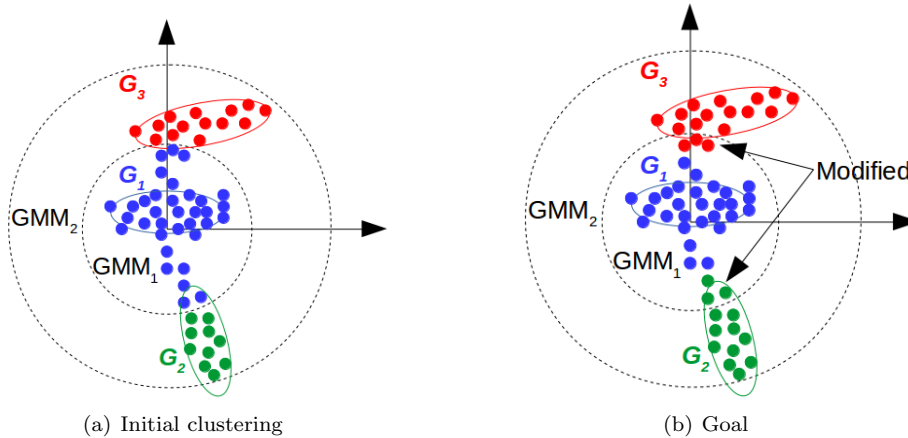


Figure 3.23: Example of pixel clustering in Gaussians

Results including spatial information

This section presents obtained results of proposed improvements for MSI and HSI. The method was evaluated using images from powders dataset [8], and University of the Negev dataset [7]. As it is described in Chapter 2, those datasets are strongly different and it allows to evaluate the capacity of the proposed model to be used in several applications. As an illustrative example, Figure 3.24 displays simplified representation of an HS image. In Figure 3.24(b) is presented the representation using only spectral data and in Figure 3.24(c) is shown the representation after including spatial channels. One may observe how the shape of the regions changes while adapting to object contours in Figure 3.24(c) as expected.

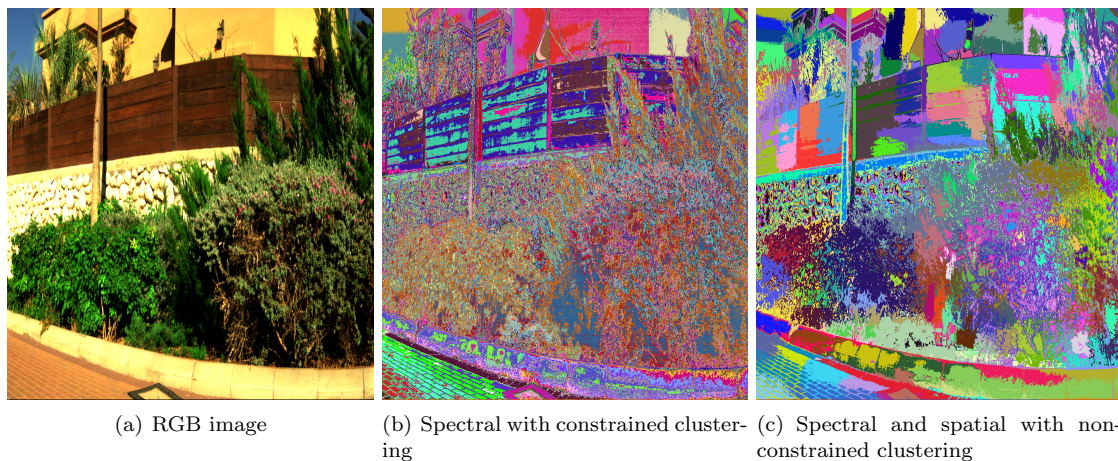


Figure 3.24: Comparison of simplified representations

Id	# Regions	<i>S</i>, <i>C</i>	<i>S</i>, <i>NO-C</i>	<i>S-XY</i>, <i>C</i>	<i>S-XY</i>, <i>NO-C</i>
1	180	28.95	<u>29.09</u>	21.77	<u>22.27</u>
2	200	29.32	<u>29.67</u>	23.25	<u>23.80</u>
3	110	26.33	<u>26.43</u>	18.61	<u>19.15</u>
4	160	29.28	<u>29.43</u>	22.69	<u>23.44</u>
5	170	29.32	<u>29.67</u>	24.21	<u>24.51</u>
6	80	22.53	22.53	23.14	<u>23.15</u>
7	90	30.08	<u>30.16</u>	30.46	<u>30.51</u>
8	90	<u>27.66</u>	27.61	28.32	<u>28.87</u>
9	60	31.68	<u>31.82</u>	28.83	<u>28.91</u>
10	80	31.71	<u>32.34</u>	33.77	<u>34.06</u>

Table 3.6: PSNR (dB) of simplified representation. ***S***: Spectral; ***S-XY***: Spectral and spatial, ***C***: Constrained to same slice, ***NO-C***: Non-constrained to same slice. Rows 1 to 5 are images from outdoors dataset [7] and the last five images from Powders dataset [8]

Table 3.6 presents PSNR values obtained in 10 images. The first five images were selected from [7] and the last five images from [8]. Simplified representations were calculated in four scenarios: ***S*, *C***: Spectral information with constrained clustering; ***S*, *NO-C***: Spectral information with non-constrained clustering; ***S-XY*, *C***: Spectral and spatial information with constrained clustering; ***S-XY*, *NO-C***: Spectral and spatial information with non-constrained clustering.

From Table 3.6, some relevant findings were identified: *i*) Non-constrained clustering of pixels in Gaussians of other slices slightly increases PSNR values compared against constrained approach. Around 0.2 dB using only spectral information and 0.36 dB after including (x, y) ; *ii*) As occurred in color images, the inclusion of (x, y) channels decreases quality of image reconstruction but increases regions compactness. Additionally, one can note how in the scenarios where improvements were proposed: inclusion of spectral information (***S-XY***) and clustering not constrained to Gaussians of same slice (***NO-C***), obtained results are better in all images.

3.5.4 Discussion about MSI / HSI

Section 3.5 presents the extension of GCL to MSI and HSI. The proposed method was tested using two datasets seeking to evaluate its performance for two different applications. In the first case, we performed some experiments using powders dataset [8] in order to build a representation that keeps the spectral diversity of the images. Obtained results demonstrate that the method preserves spectral diversity even if the number of pixels with some spectral information is small. It can be useful for those applications where the main objective is to evaluate the presence (or not) of a given spectral signature, such as material recognition or hyperspectral unmixing.

In the second set of experiments, we worked with the University of the Negev dataset [7]. In this case, the main objective was to evaluate the capability of the model to build a representation of images under uncontrolled light conditions in outdoor environments. Obtained results demonstrated that using only spectral data is not enough to build a representation with spatial coherence. It motivated us to include spatial information analogously as proposed in Section 3.4.3 for color images. After this improvement, obtained image representation preserves better contours and groups similar pixels in more compact regions. This characteristic of the proposed image representation can be

used as a preprocessing step to perform other tasks such as image segmentation or object detection.

In favor of preserving contours in image representation, we proposed some improvements to pixel clustering in GMM. As presented in Section 3.5.3, grouping related pixels in Gaussians only of their slice may affect the quality of image reconstruction in some cases. To tackle this issue, two ways of clustering were proposed: 1) Grouping pixels in Gaussians of neighbor slices; 2) Compute a single GMM using initialization values computed by slice. We performed several experiments to evaluate the modifications and we found that both of them increase the PSNR values compared to the initial version. Obtained results are presented in Table 3.6.

3.6 Conclusions

In the chapter we proposed a methodology to build simplified representations of images called Graph-based color lines (GCL) inspired by the classical Color Lines (CL) model [30]. It was developed as a generalization of the classical CL and allows to preserve spectral diversity using a combination of Gaussian Mixture Models (GMM) and a Minimum Spanning Tree. Our proposed GCL method was published in [31].

As a natural extension of the method, we introduced the model for higher-dimensional images such as MSI and HSI. Similarly, as proposed for color images, it also includes spatial information to increase region compactness and contours preservation. Obtained results demonstrate that the GCL can model small regions with a specific spectral signature, as it is shown for images from powders dataset [8].

The proposed GCL model and its variations demonstrated to be suitable for several types of images. As described with the set of experiments performed in this chapter, the method can be used not only as an initial preprocessing step to build a simplified representation of the image that preserves spectral diversity but also other tasks such as image segmentation and image dehazing. We explored the use of GCL as an over-segmentation method in the BSD 500 dataset and compare it to SLIC. Recently, in the work proposed by [133], spectral representations of images, such as GCL, are used for color correction and color grading of videos. It is an interesting perspective where our proposal can be implemented.

Chapter 4

Study of REPLICa data

4.1 Résumé

Dans ce chapitre, nous décrivons les différentes campagnes d'acquisition réalisées dans le cadre du projet REPLICa. Nous présentons l'analyse de ces données et nous concluons sur les limitations de la technologie actuelle des caméras hyperspectrales pour les scènes urbaines et dynamiques.

4.2 Introduction

In this chapter, we present the analysis of the data acquired in the context of the REPLICa project. The main goal of conducting this study was to evaluate if the inclusion of additional information to 3D point clouds improves scene description for semantic segmentation tasks. During this thesis, three acquisition campaigns including HSI, color images, and point clouds of urban scenes were carried out. The acquisitions were analyzed from different perspectives, including spectral signatures, histogram distributions, and registration errors.

4.3 Analysis of hyperspectral data

One of the main objectives of the REPLICa project was to exploit spectral information present in HSI to provide a more detailed description of the objects in urban scenes. We carried out several analyses of the acquired during this research to study different alternatives to use it. In this section, we describe the proposed methods to extract most of the relevant information of the four performed acquisitions, described in more detail in Section 2.5.2.

The data was analyzed from two complementary points of view: manual extraction of spectral signatures of some objects and studying reflectance histograms of acquired images (Section 4.3.1); by evaluating the integration of HS data into 3D point clouds, using the data acquired on Summer 2020 (Section 4.3.2). Then, the most relevant findings of the use of HS images for urban scenes are summarized in Section 4.4.

4.3.1 Spectrum and reflectance analysis

This section presents the analysis of spectral signatures and reflectance histograms of all conducted acquisitions. We followed the same evaluation protocol and extracted the spectral signatures of the same objects for each one. As described later in this section, these analyses allowed us to identify some issues in the data and partially solve them.

Autumn 2019 (V1)

This acquisition was accomplished in October 2019. Our first goal with the first REPLICIA data *delivery* was to evaluate their quality and to identify possible hints that may help to discriminate objects. We analyzed the HS data in two steps as follows:

1. Spectral signature of objects.
2. Histograms of HS images by channel.

Spectral signature of objects As the first step to analyze the data, we manually selected some objects from acquired images and studied their spectral signature. On the one hand, it was expected that spectral signatures of images acquired in the visible range with the HS03 camera were very similar to RGB images acquired with Ladybug cameras. On the other hand, spectral signatures in NIR images, HS02 camera, were expected to be similar to existing signatures of known materials in previous works, such as [15, 16].

The spectral signature of following objects was manually extracted: 1) Parking prohibited sign; 2) No entry sign; 3) Steel pole; 4) Platanaceae tree trunk, also known as Plane tree; 5) Crosswalk; 6) Grass. They were selected for three reasons: they represent different materials commonly found in urban scenes; they were close to the acquisition system and noise is considerably smaller than further objects; they were static objects.

Figure 4.1 shows RGB images and extracted spectral signatures for several areas of two well-illuminated traffic signs. Due to the characteristics of traffic signs, it was expected that spectral signatures of different regions were very different in the visible range because they represent different colors: blue, red, and white. Also, it was expected that spectral signatures of the same object were the same in the NIR range and were similar because they correspond to the same material, a sort of aluminum alloy.

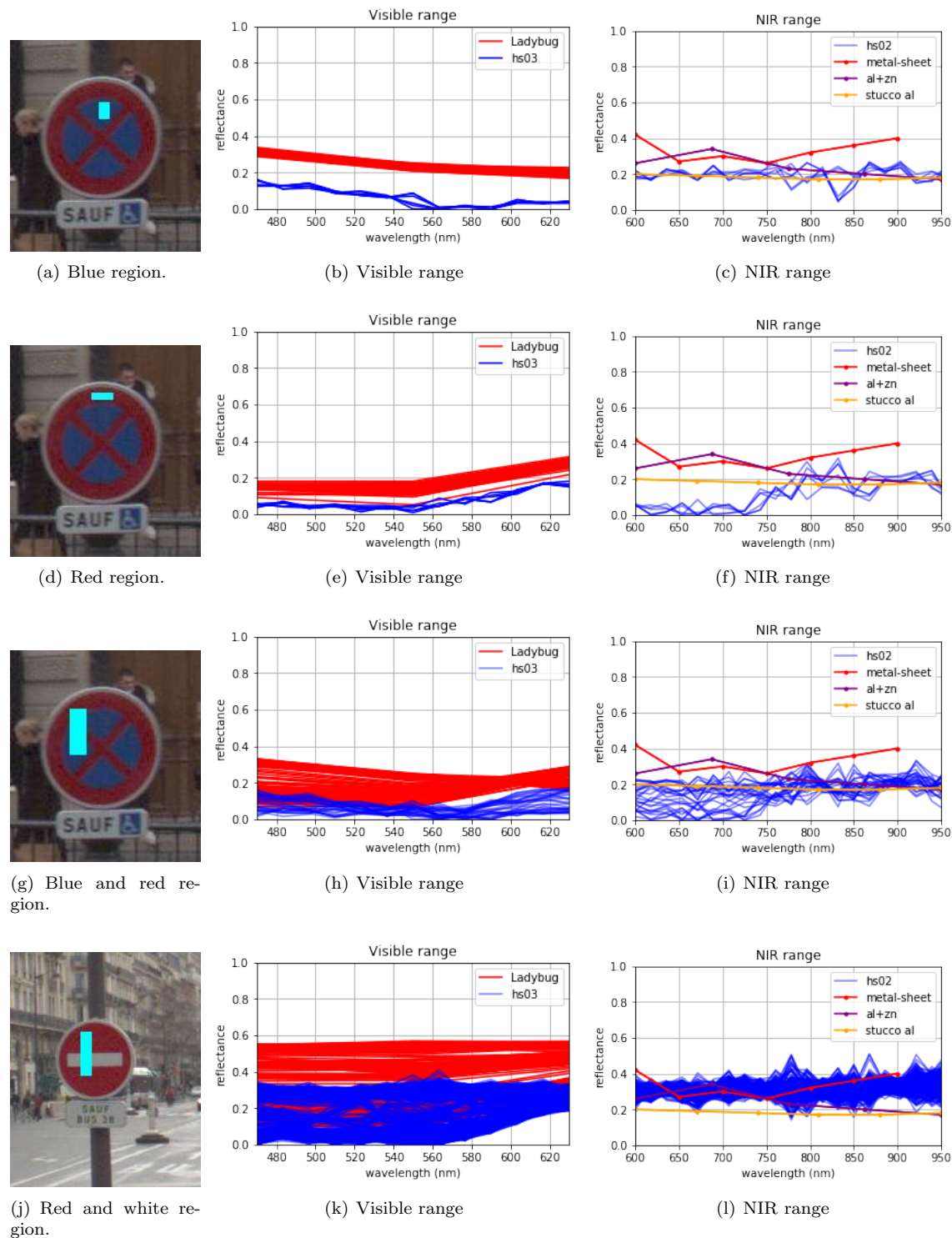


Figure 4.1: Traffic signs from the first acquisition of rue Soufflot, Paris. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from HS03 camera, in blue, and spectral signatures from Ladybug camera, in red. Third column: Spectral signatures from HS02 camera, in blue, and spectral signatures of known materials [15, 16], in other colors.

From extracted signatures of the traffic signs presented in Figure 4.1, it was identified:

- Dynamics of spectral information in the visible range is similar to the spectral signature from RGB images (middle column).
- Spectral signatures in the NIR range do not match the dynamic of some known materials such as aluminum (right column).
- Reflectance values acquired with HS cameras are smaller than those acquired with Ladybug or spectral signatures of known materials.
- Spectral signatures of the same object in NIR range vary even for pixels that are close to each other (see Figure 4.1(j), for example).
- In both cameras, but especially in HS02, variation in reflectance values is small. Most of the values are concentrated around 0.2. This last point is problematic to discriminate spectral signatures of different objects. Section 4.3.1 presents two alternatives to expand reflectance values in HS images.

The main interest to include the previous example was to illustrate the type of information obtained with HS cameras in *easy* scenarios. These cases are easy because objects are static, and lighting conditions prevented shadows that may alter spectral data. However, these characteristics are not common in outdoor environments wheremoving objects and variable light conditions are very common. Spectral signatures of the other objects on the same acquisition are presented in Figure 4.2.

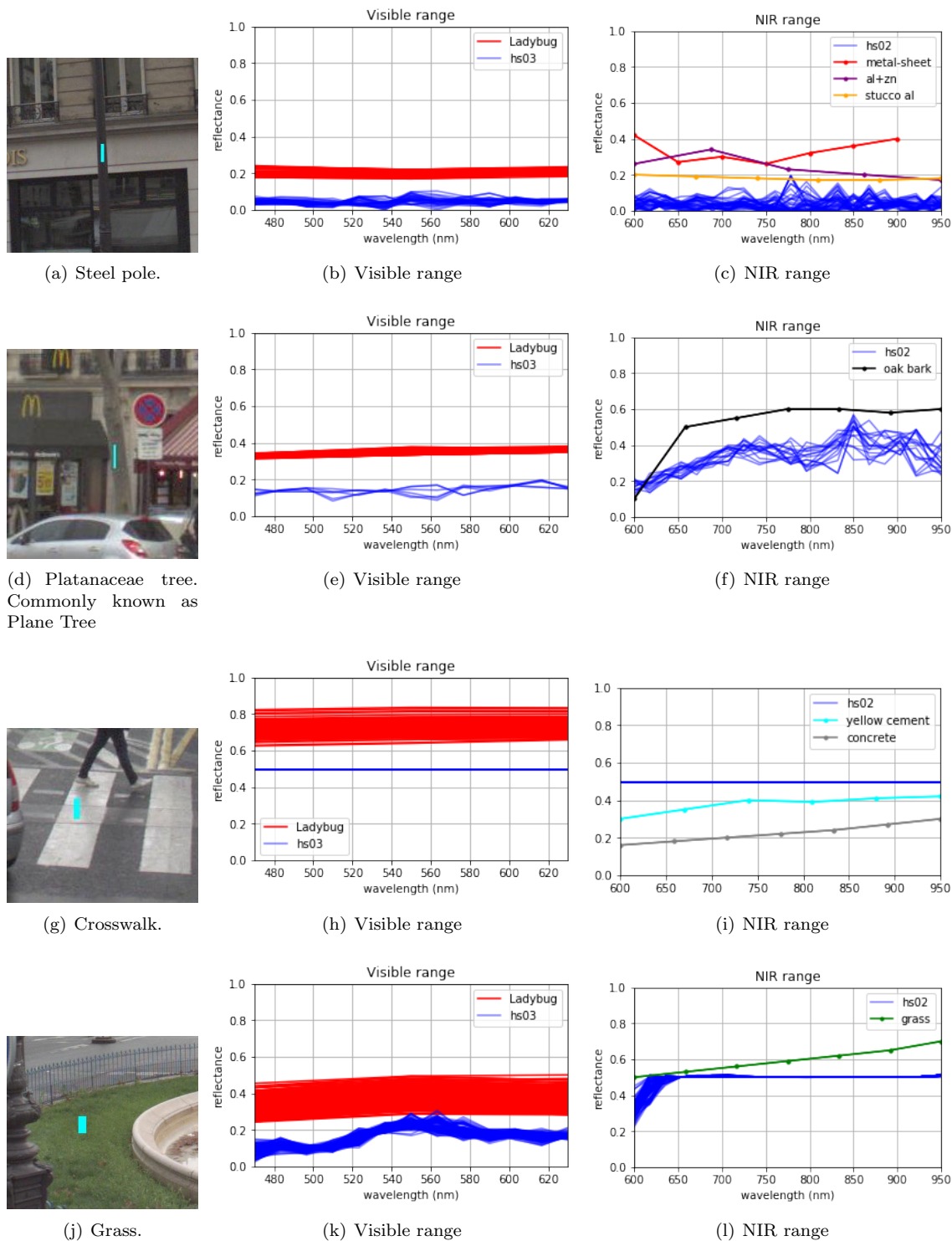


Figure 4.2: Spectral signatures of other objects from the first acquisition. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from HS03 camera, in blue, and spectral signatures from Ladybug camera, in red. Third column: Spectral signatures from HS02 camera, in blue, and spectral signatures of known materials [15, 16], in other colors.

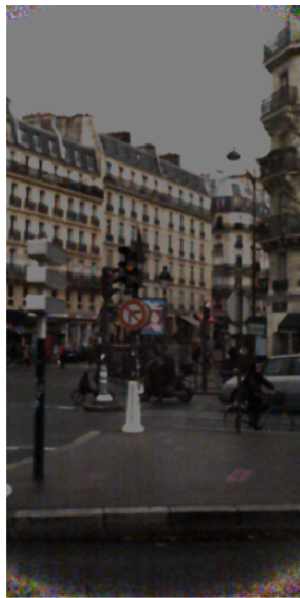
Extracted spectral signatures from different objects and presented in Figure 4.2 can be summarized as follows:

- Reflectance values in the visible range are low in all objects. If they are compared to colors of RGB images (red lines), spectral signatures of HS are much lower in all cases, making them difficult to exploit.
- Bright objects, such as the crosswalk, were expected to have higher reflectance values close to 1. Instead of that, values are over-saturated around 0.5.
- In crosswalk signatures, values are clipped or saturated in both cameras. This type of response was observed in other bright objects and in the sky.
- Values in NIR range of grass signatures are also oversaturated.
- Dark objects, such as the steel pole, have very low reflectance values in both ranges.
- In some cases, the dynamic of spectral signatures is similar to the signature of RGB images in visible range. This behavior can be seen mainly in the visible range of grass and both ranges of the Platanaceae trunk.

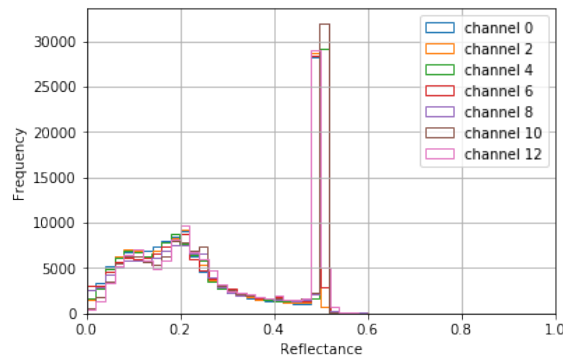
The analysis of spectral signatures in both cameras motivated us to perform more studies about HS images. Some findings, such as the over-saturation of spectral signatures close to 0.5 in some objects or the very elevated concentration of reflectance values close to zero were not as expected. The second part of the analysis, detailed in the following section, confirms an error in the postprocessing stage of the first acquisition.

Histograms of HS images Histograms are commonly used in image processing applications [134]. They represent tonal distribution, in the case of color images or reflectance values, in the case of hyperspectral images. In our case, we were focused on the understanding of the distribution of reflectance values in HS cameras and explain the identified unusual behavior in spectral signatures in Figure 4.2. The analysis presented in this section permitted us to identify an issue that occurred in the first data acquisition, particularly in the postprocessing stage introduced in Section 2.5.1.

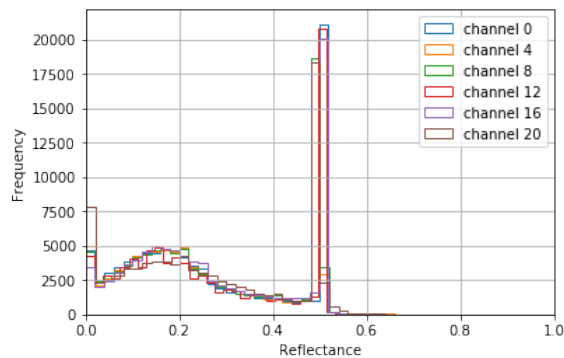
The analysis was performed by using HS images of the first acquisition of both cameras. It allowed us to identify the presence of over-saturation in all channels of both cameras, as it is presented in Figure 4.2 for crosswalk and grass images. Moreover, it was observed that most of reflectance values of all images were concentrated around 0.2 in all cases. As a manner of example, 4.3 displays the histograms of both cameras. We selected a subset of channels by image for the sake of clarity in histograms.



(a) RGB colors computed from HS image in visible range



(b) Histograms of seven channels from visible range



(c) Histograms of six channels from NIR range

Figure 4.3: Histogram of HS images from traffic-sign presented in Figure 4.1(f). RGB image was computed from image acquired in the visible range with HS03 camera.

From histograms presented in Figure 4.3, it is seen that most of reflectance values are below 0.5. This explains why spectral signatures presented in Figure 4.2(d) have smaller values than expected in both cameras compared to *reference* values. In the case of the visible range, compared to RGB images acquired with Ladybug cameras. Moreover, in the case of NIR range, compare to known spectral signatures of some materials. We computed the histograms of all images, and the same behavior was evidenced in every case: most of the reflectance values were bounded between 0 and 0.5.

In order to evaluate the distribution of reflectance values in HS images, we computed mean, minimum, maximum, and standard deviation by channel. We experimentally found that even if the data was mainly distributed below 0.5, there were also pixels between 0.5 and 1.0. Figure 4.4 shows minimum (red star), mean (blue circle) and maximum (orange circle) reflectance values by channel in all HS images. Standard deviation is presented with a vertical blue line by channel.

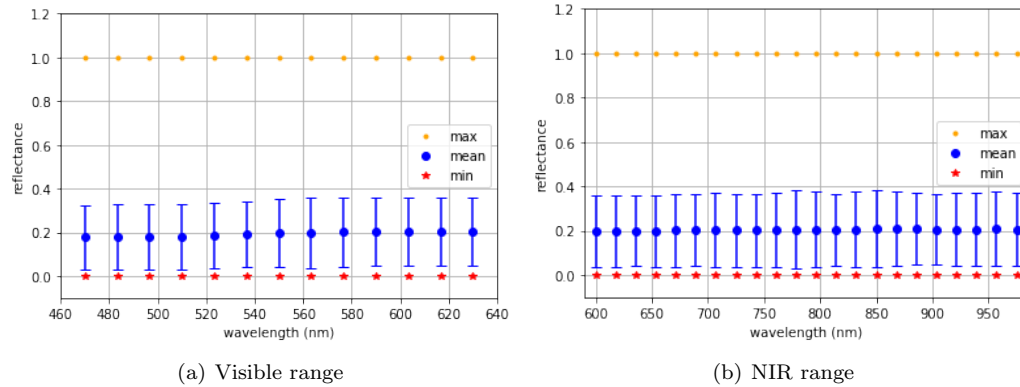


Figure 4.4: Distribution of reflectance values in HS images in first acquisition. Minimum (red), mean (blue), and maximum (orange) values are presented. Standard deviation is displayed with a vertical blue line. It is seen how the mean, minimum and maximum values of all channels are very similar in all channels of both cameras.

The analysis of the histograms presented in Figure 4.3 and then extended to all images in Figure 4.4, allowed us to identify an error in the postprocessing step of acquisition. The error had three main consequences: 1) A clip close to 0.5 in reflectance values; 2) Over-saturation of reflectance values higher than 0.5; 3) Concentration of the values in a short interval. This error strongly affected the data distribution of HS images of this acquisition. It was reported to CAOR researchers and corrected by them.

Autumn 2019 (V2)

In this section, we present the analysis of the HS images after correcting the reported error to CAOR researchers. As it is described in Section 2.5.2, the data studied in this second delivery (Autumn 2019 V2) is the same data of Autumn 2019 V2 but with the postprocessing error corrected. It allows us to directly compare spectral signatures and reflectance histograms of the first two data deliveries.

Spectral signatures The first part of the analysis took into account the spectral signatures of two traffic signs, presented in Figure 4.5. For visualization purposes, we only show the mean value of the spectral signature of pixels colored in cyan (instead of the spectral signature of every pixel as is displayed in Figure 4.1). The left column presents images with selected pixels colored in cyan; the middle column displays spectral signatures in the visible range of first and second acquisitions, and the signature from Ladybug images; the right column shows spectral signatures in the NIR

range of both cameras. In all cases, data from the first acquisition is painted in blue and data from the second acquisition in green.

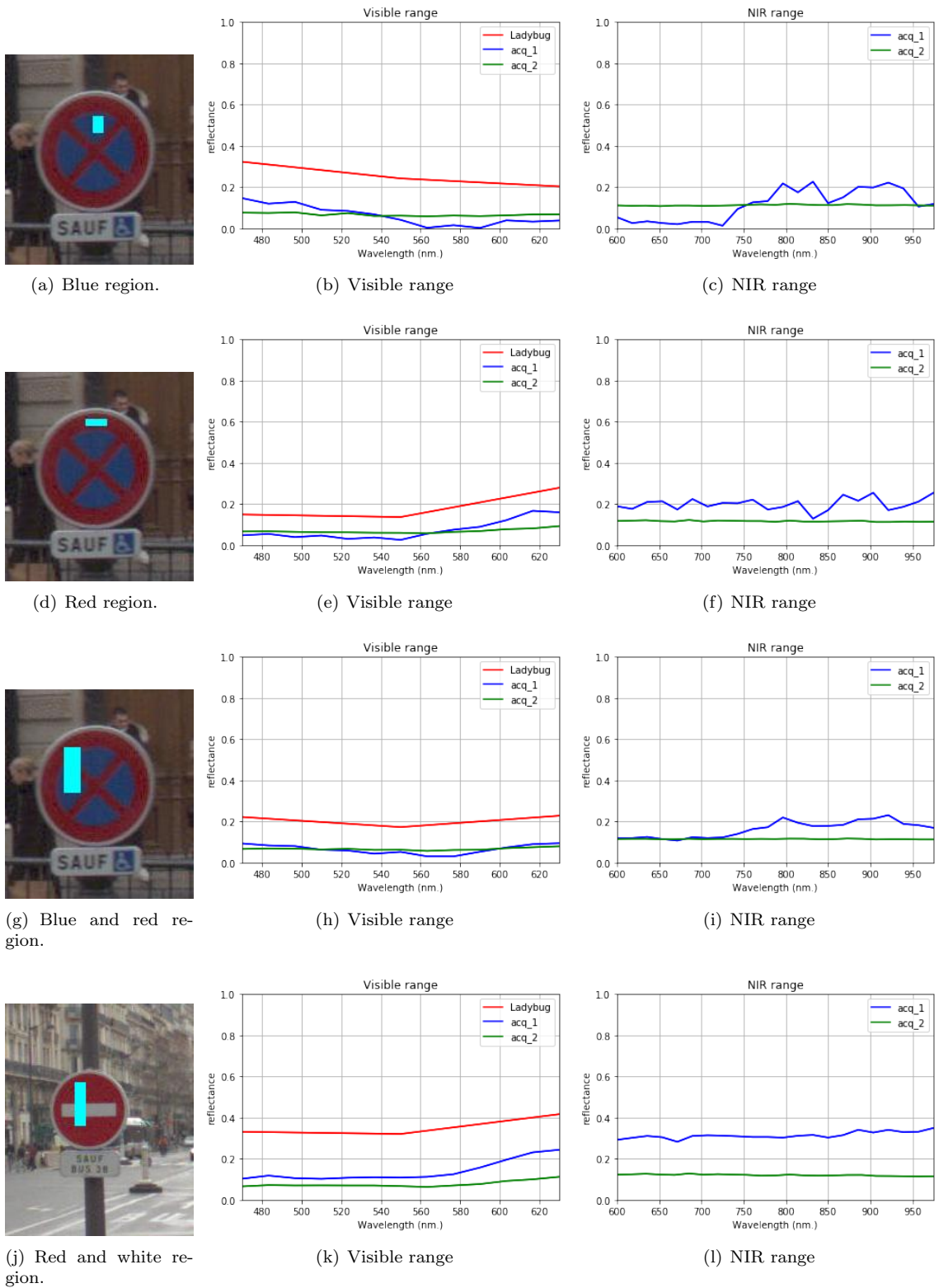


Figure 4.5: Comparison of spectral signatures of traffic sign from first (blue) and second (green) acquisitions. First column: RGB images with selected pixels to analyze in cyan. Second column: Spectral signatures from visible range camera and spectral signatures from Ladybug camera in red. Third column: Spectral signatures from NIR range.

From the results displayed in Figure 4.5, one may observe that spectral signatures of traffic signs from the second acquisition are different from previously computed ones. The most relevant findings of reflectance values are: 1) In some cases, new values are higher than previously acquired; 2) They are still smaller than RGB images; 3) Dynamics of the signal is lower than in the first acquisition, particularly in NIR range; 4) Values are systematically between 0.1 and 0.2 in both cameras. These last two findings may indicate that most of the spectral data is over concentrated. This assumption was corroborated in the second part of the analysis of the data, as described hereafter.

Histograms comparison As the second part of the analysis, we studied data distribution and histograms of the images. The main interest was to determine if the identified error that over-saturated spectral information was corrected. We computed histograms by channel of all images for each acquisition and then compared them to the first acquisition and present them in Figure 4.6.

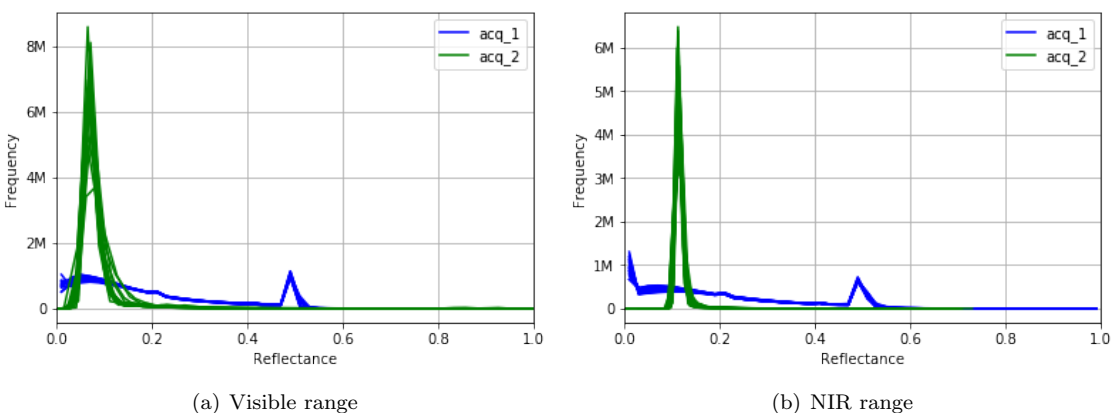


Figure 4.6: Histograms of first (blue) and second (green) acquisitions. Each line is a channel histogram of all acquired images. In both acquisitions, histograms have similar values in all channels. In the first acquisition, an over-saturation effect is present around 0.5, corrected for the second acquisition. However, spectral data is still concentrated around 0.1 in both ranges.

From Figure 4.6, it is observed that the distribution of reflectance values has relevant changes after error correction: the data are no longer saturated. However, it is even more concentrated between 0.1 and 0.2 in all channels from both cameras (up to 94 % in some cases, as presented in Tables 4.1 and 4.2). It explains the observed behavior about the low dynamics of the spectral signatures presented in Figure 4.5.

At this point, two hypotheses were proposed: 1) Reference values used to calibrate the hyperspectral cameras do not provide the *darkest* and the *brightest* values that can be found during acquisitions in outdoor scenes; 2) Illumination conditions at acquisition time (Autumn), does not provide enough light for HS cameras. In order to evaluate the first hypothesis, we propose two methods to expand reflectance values in HS images based on histogram distributions presented in Figure 4.6. To evaluate the second hypothesis, a new acquisition was performed in Summer 2020, with better illumination conditions. It is detailed in Section 2.5.2.

Expand data distribution Since the first acquisition in Autumn 2019, we observed that the reflectance values of HS images of urban environments were too concentrated. After detecting the error and its correction, it was still observed that most of the reflectance values were highly concentrated and close to 0.1. This behavior hinders the identification of different objects based purely on their spectral signature. As an alternative to make the spectral data more discriminant, we studied two methods to expand reflectance values located in a very narrow peak in all channels of images acquired with both cameras.

From histograms presented in Figure 4.6, it is seen that most of reflectance values are between 0.1 and 0.2. It implies that most of the spectral information in the images is too close to recognize different signatures. In order to measure how *concentrated* was the data, we compute the rate of pixels above different reflectance values by channel. Tables 4.1 and 4.2 present visible and NIR ranges, respectively.

Channel	0.05	0.1	0.15	0.2	0.25	0.4	0.6
1	98.6%	6.55%	3.02%	1.51%	0.86%	0.19%	0.07%
2	99.6%	11.7%	4.75%	2.63%	1.78%	0.66%	0.24%
3	99.5%	10.2%	4.55%	2.31%	1.58%	0.54%	0.12%
4	99.5%	11.5%	4.73%	2.52%	1.77%	0.65%	0.18%
5	99.3%	11.8%	4.68%	2.45%	1.71%	0.63%	0.20%
6	99.3%	11.5%	4.69%	2.52%	1.81%	0.70%	0.23%
7	98.4%	9.10%	4.15%	2.16%	1.44%	0.54%	0.13%
8	96.0%	7.34%	2.70%	1.54%	0.99%	0.34%	0.10%
9	98.4%	10.9%	3.48%	1.92%	1.28%	0.48%	0.17%
10	98.4%	15.0%	4.80%	2.52%	1.82%	0.78%	0.30%
11	99.7%	23.2%	6.54%	4.46%	2.63%	1.35%	0.64%
12	98.8%	25.7%	7.57%	4.82%	3.11%	1.57%	0.76%
13	99.8%	32.6%	10.9%	5.86%	4.47%	1.92%	0.97%

Table 4.1: Rate of pixels above different reflectance thresholds by channel in visible range.

The rate of pixels above other thresholds evidences that data is strongly concentrated. In the visible range (Table 4.1), 12 out of 13 channels have less than 10 % pixels above 0.15 and 4 % of 0.25. In NIR range (Table 4.2), data is even more concentrated: in all channels, less than 5% is above 0.1 and 0 % of 0.6. From this information, our proposals are: 1) Clip data at a given threshold; 2) Clip data at a given percentage. Then, normalize the data between 0 and 1 using the new bounding limits. The only parameter to adjust in the two methods is the value to clip the data by channel. We highlight that using these approaches, the physical meaning of reflectance is lost because original values are scaled and moved from initial reference values used during calibration.

Clip by threshold: The first approach clips the data based on a thresholding criterion. It limits the maximum reflectance value by channel according to threshold value t . Then, data is normalized to have all values distributed in the range $[0, 1]$. Figure 4.7 shows obtained results with different threshold values in the visible range and NIR range.

Normalization of reflectance values reduces the data concentration observed before, especially in the visible range. As it was expected, every value greater than the threshold value t will be assigned to t value, and consequently, a small peak in the most right part of the figures is created. However, in the NIR range, it is seen that data normalization by threshold criterion mainly moves

Channel	0.05	0.1	0.15	0.2	0.25	0.4	0.6
1	100.0%	99.9%	2.91%	1.04%	0.52%	0.07%	0.0%
2	100.0%	99.9%	3.38%	1.19%	0.62%	0.08%	0.0%
3	100.0%	99.9%	4.68%	1.69%	0.91%	0.20%	0.0%
4	100.0%	99.9%	2.90%	1.02%	0.52%	0.07%	0.0%
5	99.9%	99.7%	3.20%	1.10%	0.57%	0.07%	0.0%
6	100.0%	99.9%	4.89%	1.74%	0.93%	0.21%	0.0%
7	100.0%	99.8%	3.59%	1.23%	0.65%	0.08%	0.0%
8	99.9%	99.8%	4.50%	1.61%	0.85%	0.16%	0.0%
9	100.0%	99.9%	2.98%	1.05%	0.52%	0.06%	0.0%
10	100.0%	99.8%	2.67%	0.93%	0.46%	0.06%	0.0%
11	100.0%	99.4%	2.31%	0.77%	0.35%	0.05%	0.0%
12	100.0%	99.4%	2.63%	0.89%	0.44%	0.06%	0.0%
13	100.0%	99.9%	2.60%	0.90%	0.44%	0.05%	0.0%
14	99.9%	99.5%	1.52%	0.41%	0.10%	0.03%	0.0%
15	100.0%	99.2%	2.09%	0.67%	0.26%	0.04%	0.0%
16	100.0%	99.5%	2.28%	0.74%	0.32%	0.04%	0.0%
17	100.0%	99.9%	2.50%	0.85%	0.41%	0.05%	0.0%
18	100.0%	99.9%	2.30%	0.76%	0.34%	0.05%	0.0%
19	99.9%	99.5%	0.83%	0.11%	0.05%	0.02%	0.0%
20	100.0%	99.8%	0.94%	0.16%	0.05%	0.02%	0.0%
21	100.0%	99.5%	0.92%	0.14%	0.05%	0.02%	0.0%
22	100.0%	99.7%	0.65%	0.08%	0.04%	0.02%	0.0%
23	100.0%	99.7%	0.54%	0.06%	0.04%	0.02%	0.0%

Table 4.2: Rate of pixels above different reflectance thresholds by channel in NIR range.

the peak towards 1. It occurs because it is too concentrated around a single value. For example, in Figure 4.7(b), the data clipped at $t = 0.2$ (red lines), has most of the information contained in the interval $[0.5, 0.6]$.

Clip by percentage: Results obtained clipping by threshold do not provide enough spectral information to discriminate different materials. As a second alternative to expand concentrated data, we propose to clip data by using a percentage criterion. In this case, the procedure is slightly different, and it is performed independently by *channel*. It can be summarized as follows:

1. Compute mean value.
2. Find $p\%$ closest pixels to the mean value.
3. Compute minimum and maximum value of the closest pixels.
4. Clip input data using obtained parameters.
5. Normalize each channel between $[0, 1]$.

By using this approach, variability of spectral data is artificially increased as shown in Figure 4.8. It occurs because normalization includes the mean value by channel and allows the expansion of the closest reflectance values. For small values of p , the method will consider the closest pixels

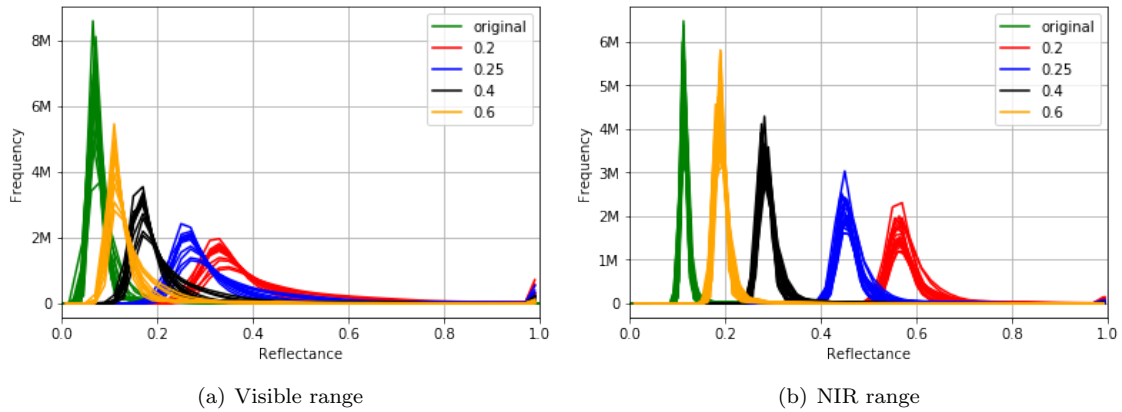


Figure 4.7: Expanded histograms based on threshold criterion. Green lines are original histograms from second acquisition without modifications. In both ranges, histograms were thresholded using $t = [0.2, 0.25, 0.4, 0.6]$.

leading to a *non-representative normalization* with over-saturated pixels close to 0 and 1 (orange line). For large values of p , most of the original reflectance values are included in the normalization, and a peak-like behavior with over concentrated data appears.

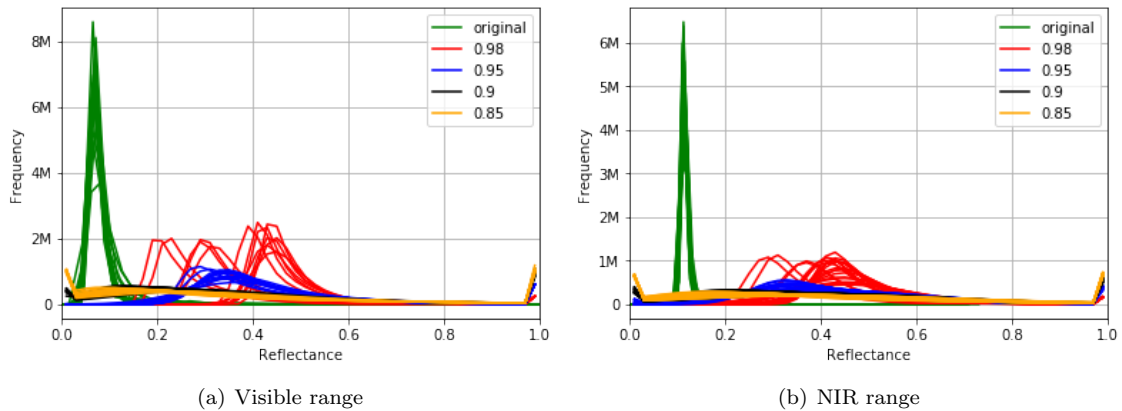


Figure 4.8: Expanded histograms based on percent criterion. Green lines are original histograms from second acquisition without modifications. In both ranges, histograms were normalized using $p = [0.98, 0.95, 0.9, 0.85]$.

Spectral signatures of normalized data In order to compare the influence of both normalization methods, we computed spectral signatures of objects presented in Figures 4.1 and 4.2. We selected $t = 0.2$ for the normalization based on a threshold value and $p = 0.95$ for the method based on closest pixels to the mean value. Figure 4.9 displays a comparison of spectral signatures

of different objects using four different reflectance values: Autumn 2019 v1, Autumn 2019 v2 and the two proposed normalization methods.

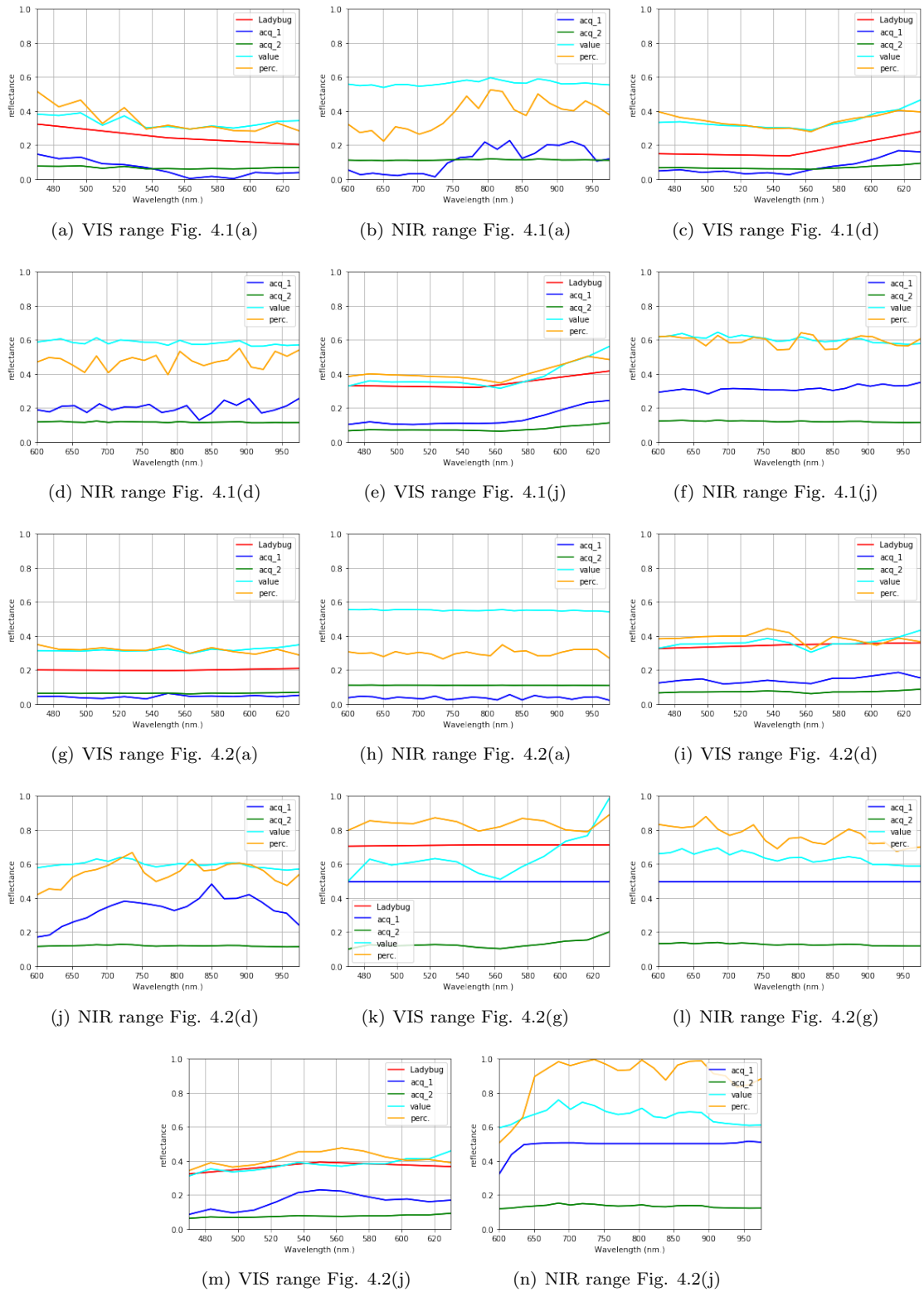


Figure 4.9: Spectral signatures using the proposed normalization methods with data from second acquisition. Autumn 2019 v1 in blue, Autumn 2019 v2 in green, normalization by threshold using $t = 0.2$ in cyan and normalization by percentage using $p = 0.95$ in orange.

Both of the evaluated methods to normalize histograms have advantages and disadvantages. On the first method based on a threshold value (cyan value from Figure 4.9), the normalization discards too bright pixels and focuses on the interval where most of the data is concentrated. However, peak-like behavior is still present, and most of the information is too close to be correctly discriminated against again. In the second method, the normalization is performed separately by channel. It allows us to enlarge the original peak in the range $[0,1]$. However, this choice adds noise to the images, and close pixels in space corresponding to the same object may have very different *reflectance* values if illumination conditions are slightly different.

An additional characteristic of normalizing the data is to lose the physical meaning of reflectance. The main consequence is that signatures cannot be directly compared to others that have not followed the same normalization process. It can be critical, especially for the second method, because normalization is performed independently for each channel and depends on its reflectance values. In Figure 4.9, one may observe that spectral signatures change not only in dynamics but also in the range of *reflectance* values according to the type of normalization. Additionally, we found that in some cases (Figures 4.9(b), 4.9(d), 4.9(b), 4.9(e) and 4.9(h)), the results of normalization based on percentage (orange) are very similar to those of Autumn 2019 v1 (blue) in NIR range.

This last finding of the similarity of the normalization results and the data of the first acquisition is relevant. First of all, because it shows that normalizing the data may lead to the error identified in the first acquisition. And second, because due to the correction of the reported error, the data of the second acquisition was even more concentrated. At this point of the work, we identified that the raw data acquired by HS cameras of the first acquisition, aside from the postprocessing stage or the normalization methods, was not appropriate to work with: reflectance values are too close around the same value in all channels without providing discriminant spectral signatures between different objects. This finding guided us to perform the third acquisition in the summer of 2020, expecting to have reflectance values distributed over the whole range, thanks to better illumination conditions.

Summer 2020

This acquisition was conducted in June 2020, and it is described in more detail in Section 2.5.2. The main motivation was to study if sunny days provide enough light to HS cameras to improve reflectance values. The same protocol to analyze the data used in previous acquisitions was followed by using spectral signatures and histograms.

Spectral signatures This acquisition was conducted following the same path that in the first acquisition. It allowed us to manually extract spectral signatures of the same objects and directly compare them to previous ones. Figure 4.10 shows seven signatures of the objects presented in Figures 4.1 and 4.2.

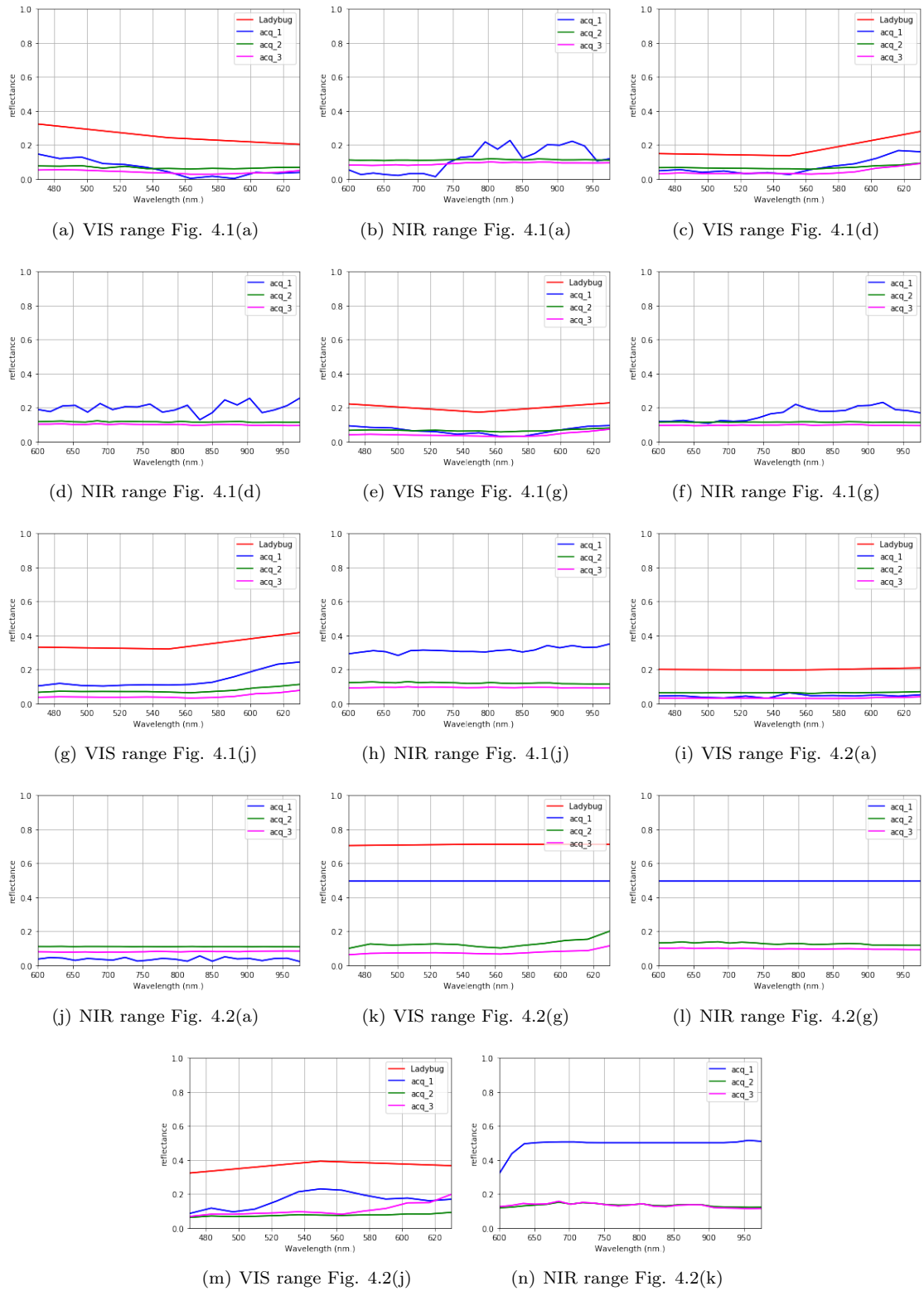


Figure 4.10: Comparison of spectral signatures of same objects in first three acquisitions. First acquisition in blue, second acquisition in green and third acquisition in green.

The spectral signatures presented in Figure 4.10 demonstrate that low reflectance values obtained in first acquisitions were not caused only by meteorological conditions. In most cases, magenta lines (third acquisition) are below green lines (second acquisition) in both ranges. The only case where it was observed an augmentation of the reflectance was in the case of the grass signature, presented in Figures 4.2(j) and 4.2(k).

Histograms The analysis of histograms was carried out by selecting a subset of 92 of 547 available images, equally distributed over the data. Figure 4.11 displays the histograms of the images from the first three acquisitions. Obtained results show that reflectance values are still low in the third acquisition and concentrated close to zero in both ranges, as occurred with previous acquisitions. Moreover, it is seen that in the NIR range, values are even lower than previous ones.

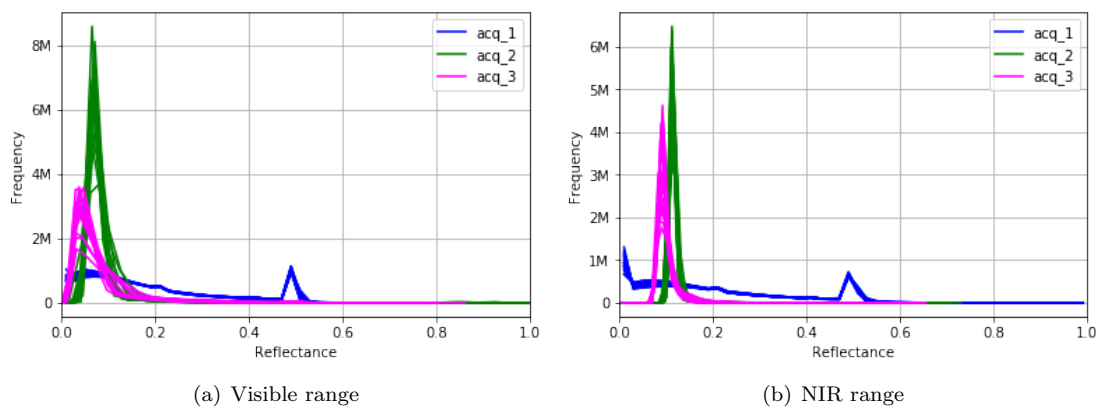


Figure 4.11: Histograms of first (blue), second (green) and third (magenta) acquisitions. Each line is a channel histogram of all acquired images. In the first acquisition, an over saturation effect is present around 0.5. The second and third acquisitions have same post-processing stage. One may observe that even if the last acquisition was conducted in summer, reflectance values are still low in all channels. In NIR range, they are even lower than in previous acquisitions.

The results of the third acquisition proved that the second hypothesis: “*Illumination conditions at acquisition time (December 2019, end of the autumn), does not provide enough light for HS cameras*” was not correct. We carried out two acquisitions of the same region under very different light conditions and the spectral data was very similar. In the context of the REPLICA project, the main interest of acquiring HS images was to recognize different materials based on their spectral signature in the NIR range. Conducted analyses of the acquisitions demonstrate that data is too concentrated on very narrow peaks that cover the low range of reflectance. With the aim to establish if those narrow peaks contain relevant information for materials recognition, a fourth acquisition was carried out.

Static summer 2020

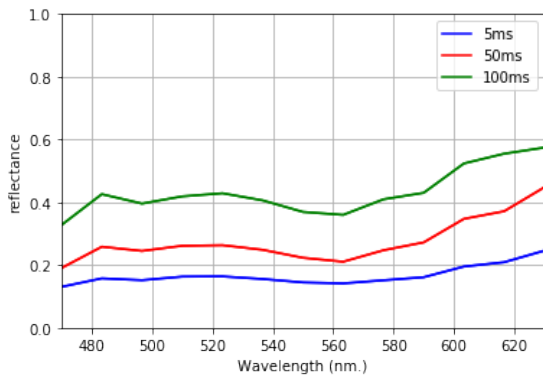
This acquisition was performed in a static scenario using only the HS03 camera. Figure 2.24 shows acquired images in the parking of MINES Paristech university. In this case, our goal was to evaluate

the influence of exposure time with respect to the quality of spectral information. Consequently, images were acquired one after the other on the same day, varying only the exposure time between 5ms, 50 ms and 100 ms. The analysis was carried out by using spectral signatures and reflectance histograms, as before.

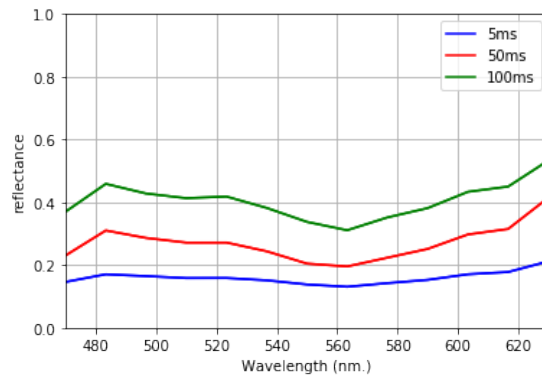
Spectral signatures Spectral signatures of four objects were manually extracted: asphalt, wood table, grass, and tree trunk. Figure 4.12 shows the region where each signature was extracted as well as a comparison of the spectral signatures varying exposure times of the camera.



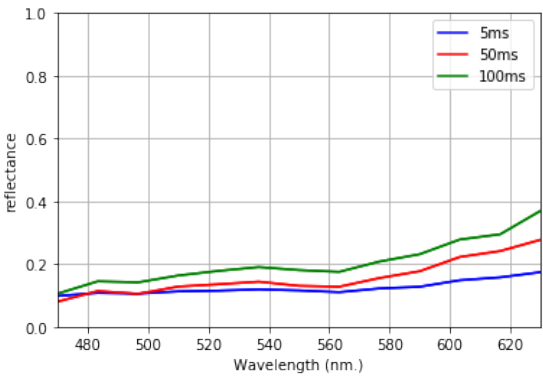
(a) Image acquired with exposure time of 100 ms in visible range and selected objects to extract spectral signatures



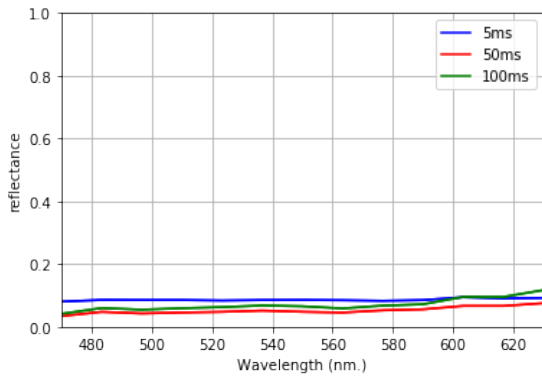
(b) Concrete



(c) Wood table



(d) Grass



(e) Tree trunk

Figure 4.12: Spectral signatures of same objects varying exposure time between 5ms (blue), 50ms (red) and 100ms (green) of fourth acquisition.

Extracted signatures with higher exposure time have larger reflectance values and higher dynamics in three of four studied objects. This is an expected result because the exposure time defines the duration that the digital sensor inside the camera is exposed to light. The use of longer exposure times permits the acquisition of more relevant information by the camera about the environment. However, if there are moving objects on the scene (or in scanning system), it may lead to a blurring effect.

Histograms The second part of the analysis used reflectance histograms for each image. In Figure 4.13 is presented a comparison between 5 ms (green), 50ms (red) and 100ms (blue) exposure times of acquired images. In this case, it is seen that images acquired with larger times have reflectance values less concentrated. For example, for 100 ms (blue), histograms of some channels have even a small peak around 0.5 that is not present in the other cases. For 5ms (green), the peak-like behavior present in the summer 2020 acquisition appears again.

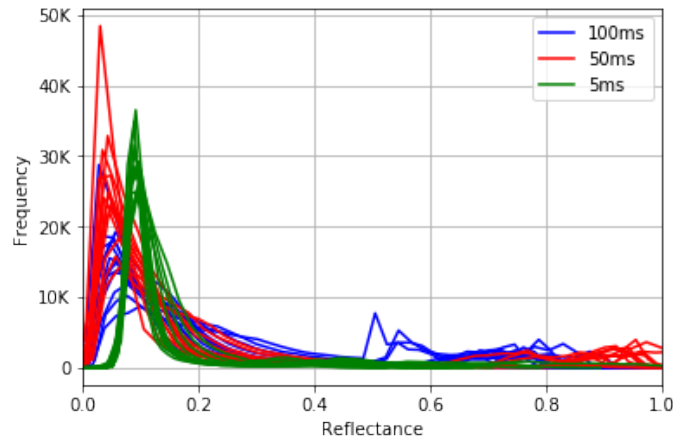


Figure 4.13: Histograms of images acquired in fourth acquisition with different exposure times: 100ms (blue), 50ms (red) and 5ms (green).

The results obtained in this last acquisition demonstrate that exposure time plays a critical role in hyperspectral images. In this scenario, there are no moving objects in the scene and the acquisition system was static. It allowed us to increase the exposure time of the camera without creating a blurring effect. Neither of the two previous conditions can be guaranteed to improve the acquired images for urban street applications. At this point of the analysis, we conclude that the current technology of HS cameras is not capable to perform acquisitions in outdoor scenes where objects (traffic-actors and acquisition system) are in constant movement.

4.3.2 Integration with point clouds

Additionally to the analysis conducted in the spectrum of the HSI, we also studied the impact of their integration into 3D point clouds approaches. We projected HS images into point clouds using the transformation matrices between the LiDAR sensor and each HS camera to assign spectral values to 3D points. The main goal was to evaluate the offset between 2D and 3D data due to

registration errors. This is a common issue when data is acquired using different sensors. As presented in Section 2.5.1, the acquisition system is composed of several sensors including a GPS, an IMU, a LiDAR sensor, color, and hyperspectral cameras.

We selected a dense point cloud from the third acquisition and manually segmented some objects using CloudCompare software: five traffic signs and a trunk tree. Then, we projected them onto the closest HSI based on timestamp information. As a manner of example, Figure 4.14 displays a crop of the selected point cloud with the points projected in the NIR camera (brown) and manually segmented traffic signs (green, orange, and red).

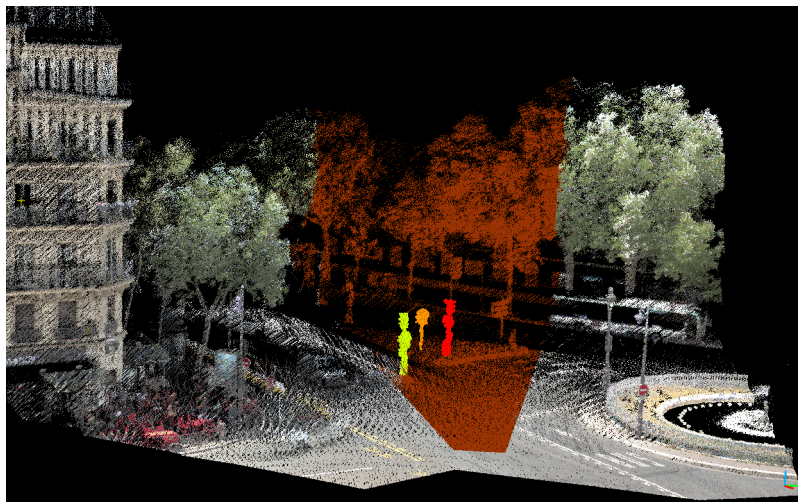


Figure 4.14: Crop of selected point cloud to study the offset between 2D and 3D data. The points seen in the same HS image are painted in brown and three of the manually segmented traffic-signs in green, orange and red.

The point cloud presented in Figure 4.14 was selected for the analysis for two reasons: few moving objects and a field of view of HS cameras with close and far objects. The former to prevent occlusions and the latter, to evaluate the influence of distance the registration between 2D and 3D data.

The selection of the HS image to project 3D segmented objects was made manually. It was a simple procedure divided in: 1) Extract timestamp of the segmented object from PLY file of the point cloud ;2) Manually search the images where the segmented object is present; 3) Select image with the most similar timestamp. In some cases, one of the acquired images has a close enough timestamp to the 3D data timestamp, and the color image is then correctly registered (see building or tree in Figure 4.14, for example). However, it is not always the case. Figure 4.15 shows the HS images and the projection of 3D points of segmented objects. The selection of the closest HS image can be done automatically. However, because of the very low quality of the spectral information, we performed this stage manually.

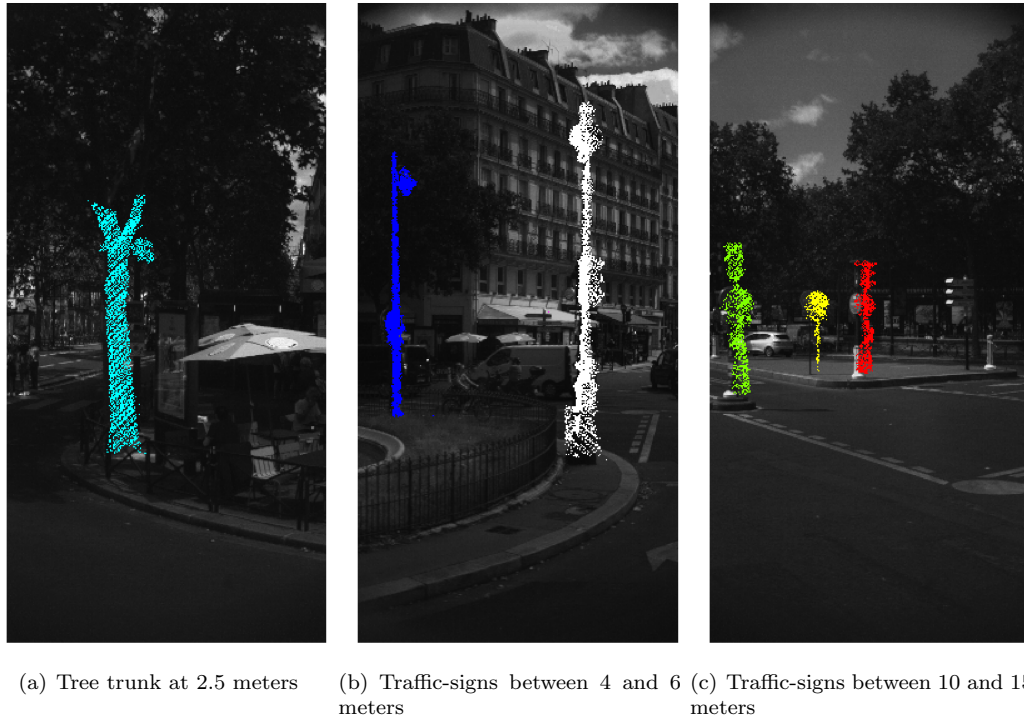


Figure 4.15: Projection of 3D points into the closest HS image based on timestamp information. Images a), b) and c) are not the same.

The results obtained with manually segmented objects show a significant offset between data acquired with LiDAR sensor and HS cameras, especially for far objects. This mismatching mainly occurs for two reasons: miss-synchronization between the different sources of data; movement of the acquisition system and the objects. The first problem is very common in these types of applications and, in our case, it was tackled by triggering every sensor of the acquisition system simultaneously. The obtained RGB colored point clouds demonstrate a valid solution to align color images and 3D point clouds in most cases. The second cannot be directly tackled because mapping systems in urban scenes are expected to face dynamic environments. Increasing the exposure time of HS images will lead to higher offsets between 2D and 3D data because of the movement of objects and the acquisition system.

4.4 Most relevant findings

The analysis of the four acquisitions of HSI demonstrates that the use of this type of data for mobile mapping applications is still an open research subject. The most relevant findings are summarized as follows:

1. **Cameras calibration:** Most MS/HS cameras need a calibration step before every acquisition. It requires a black target and a white target, representing the darkest and the brightest

values found in the scene. They are used as the reference values in order to acquire reflectance values bound in this range. In the first three acquisitions (Autumn 2019 v1, Autumn 2019 v2, and Summer 2020), it was observed that most of the reflectance values were highly concentrated in a short reflectance interval. It points out that the reflectance range given by the used target (*provided by the manufacturer of the cameras*) values is not representative enough for the type of objects and illumination conditions present during the acquisition in outdoor scenes.

2. **Low reflectance values:** In the first three acquisitions, it was observed that reflectance values were very low. In some cases, such as black objects that absorb most of the light, it is expected behavior. However, we found this behavior in every channel of every image, where most of the reflectance values were concentrated around 0.1. Three alternatives to improve low reflectance values were studied: 1) Acquire new data under better lighting conditions (summer instead of autumn); 2) Expand data by analyzing the histograms; 3) Increase exposure time of the cameras. The first two methods did not provide satisfactory results. In the third one, in a static environment, reflectance values were improved.
3. **Moving objects:** The dynamic environments of urban scenes avoid increasing exposure time. Not only by the presence of moving objects in the scene: cars, pedestrians, bikes, buses but also because the acquisition system is mounted in a moving vehicle. It was experimentally found that the use of values higher than 5 ms leads creates a blurring effect.
4. **Integration with point clouds:** The projection of point clouds in HS images and vice versa displays a spatial offset between the data. It is even more visible when objects are further from the sensors. The main reasons for this offset are: miss-synchronization of different sensors; dynamic objects in the scene. A consequence of increasing the exposure time to improve reflectance values will lead to higher offsets in moving objects.

4.5 Conclusions

In this chapter, we presented the analysis of acquired hyperspectral data in the context of the REPLICA project. It was carried out in two stages: first, by studying their reflectance histograms and spectral signatures of different objects; second, by evaluating the projection of color data on 3D data and vice versa. The first part of the analysis was conducted with the four acquisitions, previously described in Chapter 2. Several issues were identified, including the calibration of the darkest and brightest reference values, the presence of low and very concentrated reflectance values, and the limitation to increase the exposure time of cameras due to dynamic environments. The second part of the analysis was performed to study the integration of HS data in 3D point clouds. We manually segmented some 3D objects, including poles and tree trunks, and then projected them into the images. An offset between the projected points of the poles and the poles present in the images was observed. In the cases where the poles are close to the sensors, the offset was slight. However, for the further objects, it considerably increases.

The findings of the analysis of HS images demonstrate that the use of this type of data for mobile mapping applications is still an open research subject. Some points are questions to be studied for the research community, such as:

1. How to improve acquired reflectance values without increasing the exposure time in dynamic environments? With the current technology, the highest exposure time without creating a

blurring effect was 5 ms. However, as we present in Section 4.3.1, higher reflectance values lead to more discriminant spectral signatures.

2. Can we reduce the offset between HS images and 3D data with shorter exposure times? In carried out acquisitions, exposure times were increased as much as possible to improve spectral information. However, in dynamic scenarios, further objects are even more impacted by the registration error (see Section 4.3.1).

As a complementary study of the spectrum of HSI presented in this chapter, we evaluated the influence of RGB information in point clouds segmentation. We did a comparative analysis under different training scenarios and presented it in Section 6.6. Obtained results demonstrate that the inclusion of additional information to 3D point clouds permits the improvement of the models in urban scenes.

Chapter 5

Power Jaccard losses for semantic segmentation

5.1 Résumé

Dans ce chapitre, nous proposons une nouvelle fonction de perte appelée “Power Jaccard loss” pour les tâches de segmentation sémantique. Nous avons évalué notre contribution en utilisant différents types de données telles que des images en niveaux de gris et en couleurs et des projections 2D de nuages de points. Les résultats obtenus démontrent que notre contribution peut être appliquée à un large éventail d’applications, y compris les images aériennes, les chiffres, les scènes urbaines et le phénotypage des plantes.

5.2 Introduction

Loss functions play a crucial role in optimization algorithms because they quantify the quality of the model into a single number. It permits the conversion of the learning problem into an optimization problem where the loss function must be minimized. In this chapter, we introduce our novel power Jaccard loss [32] for semantic segmentation tasks. We studied its influence under different scenarios to demonstrate its pertinence and most relevant advantages in Deep Learning models.

The content of this chapter is divided as follows: Section 5.3 describes main stages of the process to train a Deep Learning (DL) model; Section 5.4 introduces loss functions and their relevance in image segmentation tasks; Section 5.5 presents our proposed Power Jaccard loss; Sections 5.6 and 5.7 displays experimental design and results with several datasets with our proposal; then, discussion and result analysis are presented in Section 5.8. Finally, conclusions are stated in Section 5.9.

5.3 Overview of DL

Deep Learning (DL) techniques have demonstrated an enormous impact in the last decade in most of the fields of knowledge, including image processing, bioinformatics, recommendation systems, natural language processing, automatic speech recognition, point clouds analysis, among others

[135]. These techniques have been widely studied, and several works can be found with a detailed explanation, such as [136, 137].

The process to deploy a deep learning model can be summarized in:

1. *Architecture definition:* According to the type of problem to be solved (e.g. image segmentation, text translation, anomalies detection), a different architecture must be chosen. For example, in image processing applications, U-Net based architectures [138], based on convolutional neural networks are commonly used for semantic segmentation.
2. *Model compilation:* After model architecture selection, it is required to choose a set of parameters that will rule the learning stage. Some of those parameters are loss function, optimizer, batch size, and learning rate. Parameters selected in this step will strongly affect the overall performance of the model.
3. *Model training:* It contains all operations related to data preparation such as filtering, normalization as well as their splitting in the train, validation and test sets. The training stage is performed until a criterion is fulfilled, such as the stop of improvement in the validation set or the reach of the maximum number of epochs.

One of the most critical points during training is to prevent the model from overfitting training data. There are several ways to do it, such as early stopping, data augmentation, regularization, and dropout. A more detailed explanation of techniques to prevent overfitting is presented in [139].

4. *Evaluation:* The evaluation of the model must be performed with data that has not been seen before. If the volume of available data is too small to be separated into three subsets without bias, one alternative is to use cross-validation.

The evaluation of the model should be carried out, if possible, with several sources of data. It will allow not only to obtain a more reliable performance score but also to identify potential bias linked to the data used during training. For example, a model trained for semantic segmentation tasks using Cityscapes dataset [4] should perform well with color images from urban scenes of European cities.

5. *Deployment:* The last stage of the model is not always implemented. Usually, it is included in industrial applications where it is expected to make it available for more people. Cloud services such as Amazon Web Services (AWS) or Microsoft Azure provide some alternatives to implement it. Also, DL models can be integrated on embedded systems.

The five steps presented before are usually part of an iterative process. In general, the first three stages can't be performed and well-tuned in a single run. It implies that deploying a deep learning model is a looping process, where the most critical part occurs during training. In Figure 5.1, we present a diagram of the training stage of a Convolutional Neural Network (CNN) in a semantic segmentation task. We emphasize that it is a simplified representation of the training stage, and according to the application and the model architecture, additional steps can be included.

The diagram of Figure 5.1 divides the training stage into two parts: forward step and backward step. In the first part, a forward pass of the input data over the network is performed and a prediction is obtained. In the case of semantic segmentation for images, the outputs of the model are class membership probabilities per pixel. In the diagram is presented the probability of membership to "vehicle" class by pixel. It can be seen that the highest activation values are on the back of the

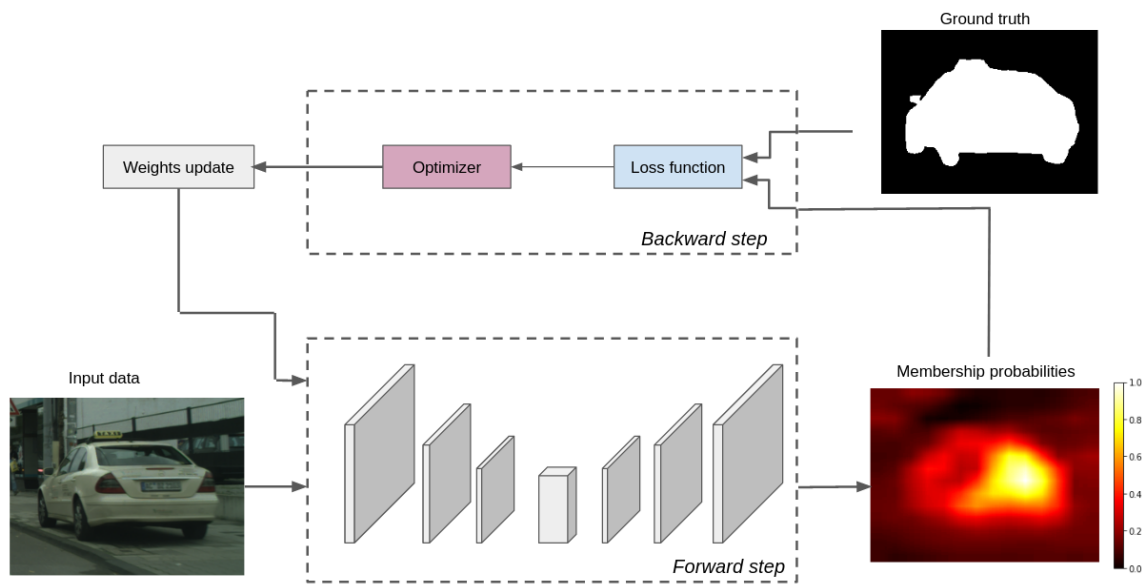


Figure 5.1: Simplified diagram of training stage of Convolutional Neural Network (CNN) in a semantic segmentation application.

car. Also, it shows that foreground pixels, such as the road or the wall on the top of the image, have lower probability value .

The second part of the diagram is the backward step, and its primary goal is to update model weights. In this step, two inputs are required: the output of the model from the forward step and the ground truth of input data. In this example, one may observe that most of the pixels that actually correspond to the car have high activation values. Even though it can also be seen that probabilities are slightly lower close to the center of the car. It may indicate that the model's prediction was incorrect for specific pixels. To solve these errors, the parameters weights of the model should be updated.

The process to update model weights relies on error calculation between predicted and expected labels and their propagation over the layers. Error is computed using the loss function, also known as Cost Function, representing prediction error as a real value. Then, the error computed at the output of the model is propagated backwards (a.k.a. backpropagation algorithm) over the layers, and parameters weights are updated. This propagation is performed employing an optimization algorithm, most of the time based on gradient calculation. A more detailed definition of loss functions, as well as their importance in statistical learning, is presented in [140].

Some other parameters also play an important role during training, such as batch size, learning rate, activation function, and momentum. In our case, we were mainly focused on the loss function for semantic segmentation applications in images and point clouds. Results obtained directly working with 3D data are presented in Chapter 6.

5.4 Loss functions

Semantic segmentation using learning-based approaches is an active research topic. One of the most common issues in this task is related to highly unbalanced datasets. Several strategies have been proposed to compensate less populated classes. They can be mainly clustered in two categories: 1) Data-level methods, increasing artificially the number of training samples via *data augmentation* through over-sampling and under-sampling training samples; 2) Algorithm-level methods, without modifying the training data distribution, the decision process increases the importance of smaller classes [141]. In this thesis, we focus on the second approach by modifying the loss function to penalize model mistakes similar to focal loss [142].

According to [143], loss functions can be mainly divided into two groups: **1) Statistical-based** such as Cross-Entropy (CE) and some of its variants such as Weighted CE [138], distance map penalized [144] that computes a mask based on pixels that are close to a given class, Focal loss and top K-loss [145] that drop pixels when they are too easy to classify, given a threshold parameter. **2) Geometrical-based** loss functions inspired by discrete sets and mostly motivated by Sørensen-Dice score. As an extension of Dice, Tversky loss [146] allows to penalize differently False Positives (FP) and False Negatives (FN); well known Jaccard loss or Intersection over Union (IoU) [147] and its multiclass version *mean IoU*; boundary loss [148] takes the form of a distance metric in the space of contours. Penalty Generalized Dice (pGD) [149] seeks to penalize with an additional parameter both FP and FN. *Geometrical-based* loss functions are an active research field with successful results in semantic segmentation [150].

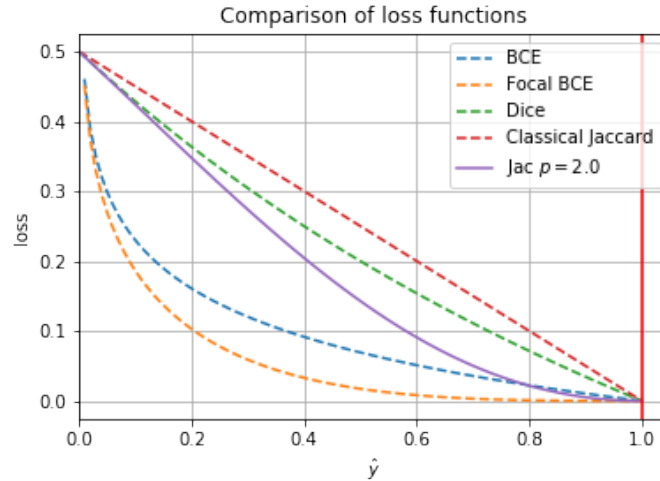


Figure 5.2: Comparison of some classical loss functions (dotted lines) and our proposed Power Jaccard loss (solid line). Binary Cross-Entropy. For Focal BCE with $\gamma = 2$. Vertical red line indicates the ground truth $y = 1$. Our proposal reduces the relative loss for well-classified examples.

5.5 Power Jaccard loss

In this section, we introduce our proposed “Power Jaccard loss” to perform semantic segmentation tasks [32], as a generalization of the Jaccard loss function. In the proposed loss, the higher the power term, the stronger the penalization of the worst predicted samples. We have evaluated our proposal with several segmentation datasets such as MNIST, Cityscapes [4], Street 3D dataset [10] and aerial images [6]. The use of power losses improves the performance in binary and multiclass segmentation. Figure 5.2 illustrates a comparison between the proposed loss functions and other classical losses as cross-entropy, Jaccard, and Dice score. The abscissas represent the predicted value \hat{y} and the ordinates the corresponding loss value.

We will see that a higher value of p in our generalized Jaccard loss function improves model convergence by shifting the focus to improve harder predictions far from one value.

5.5.1 Jaccard index

Our proposed “Power Jaccard loss” is a generalization of the well-known Jaccard index, introduced in [151]. It measures the similarity between finite sample sets A, B as the Intersection over Union (IoU): $\frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$. The Jaccard index is zero if the two sets are disjoint and is one if they are identical. Other similarity indices exist such as *Dice’s index* defined as $\frac{2|A \cap B|}{|A| + |B|}$ and it can be rewritten in terms of Jaccard as $\frac{2J}{1+J}$. For minimization purposes, it is recommended to use the *Jaccard distance* $J_d = 1 - \frac{|A \cap B|}{|A \cup B|}$ a.k.a. Steinhaus distance or biotope distance, proposed to compare unordered sets [152]. For image segmentation, a differential version is commonly used [147]. Furthermore, Jaccard distance has been successfully implemented as a loss function in deep learning models due to its independence to class imbalance [153].

During the segmentation process, the loss function should evaluate each pixel i measuring the distance between its ground truth $y_i \in \{0, 1\}$ and the current result of the model \hat{y}_i , the estimated probability value representing its likelihood of being part of the object. Subscript i is removed for simplification reasons in y_i and \hat{y}_i . The *straightforward* implementation of J_d as a loss function in continuous domain replaces intersection and union by product and sum as follows ([154] and [155]):

$$J_1(y, \hat{y}) = 1 - \frac{(y \cdot \hat{y}) + \epsilon}{(y + \hat{y} - y \cdot \hat{y}) + \epsilon} \quad (5.1)$$

where ϵ prevents zero division.

In [156] is introduced a variation of J_d , adding a power term to the inner product that measures the distance between density probability functions. Thus, a power term equal to 2 is included in the denominator:

$$\begin{aligned} J_2(y, \hat{y}) &= 1 - \frac{(y \cdot \hat{y}) + \epsilon}{(y^2 + \hat{y}^2 - y \cdot \hat{y}) + \epsilon} \\ &= \frac{(y - \hat{y})^2}{(y^2 + \hat{y}^2 - y \cdot \hat{y}) + \epsilon} \end{aligned} \quad (5.2)$$

This modification from (5.1) to (5.2) can be interpreted in the context of *focal loss*, where the main idea is to reduce both loss and gradient for correct prediction while emphasizing the gradient of errors (See Figure 5.2). The implementation of the binary loss presented in Equation 5.2 for

the multiclass case is straightforward using *one-hot encoding* for model's output and ground truth labels.

5.5.2 Our proposal: Power Jaccard Loss

We propose a generalized loss function called *Power Jaccard Loss* including a power term p to the Jaccard loss of (5.1) to increase the weight of wrong predictions during training, as follows:

$$J_p(y, \hat{y}, p) = 1 - \frac{(y \cdot \hat{y}) + \epsilon}{(y^p + \hat{y}^p - y \cdot \hat{y}) + \epsilon} \quad (5.3)$$

If $p = 1$, our proposed loss is identical to Jaccard distance. Previous works have directly used $p = 2$ in geometrical losses such as Dice score [?], Jaccard distance [157] and Tanimoto distance [158, 156].

Figure 5.2 illustrates the shape of loss functions according to p . We propose to increase the weight of wrong predicted samples depending on p . Also, as shows Figure 5.3 for $p > 2$, the minimum of loss function is not at $\hat{y} = 1$. It implies that the model will converge to a non-desired optimal value, and negative loss values would be obtained. We also demonstrate that p must be between one and two. We present an analytical explanation about negative values and derivatives in Section 5.5.3.

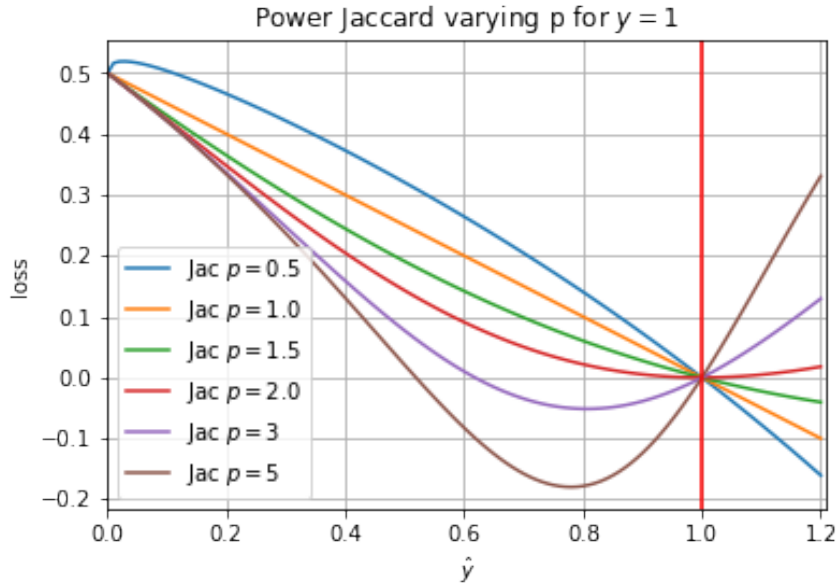


Figure 5.3: Incidence of parameter p in power Jaccard loss. Vertical red line indicates the ground truth value of $y = 1$.

5.5.3 Derivatives of Power Jaccard

We studied derivatives of the proposed Power Jaccard to demonstrate the influence of p value during learning. In this section, we demonstrate that we must have $1 < p \leq 2$ to converge to a minimum located between 0 and 1.

As $y \in \{0, 1\}$, the power term p does not affect y value. Eq. 5.3 can be rewritten as presented in Eq. 5.4. In order to find the minimum value of the loss function, we compute $\partial J_p / \partial \hat{y}$ and equated to zero. We recall that $\hat{y} \in]0, 1[$ because of the activation function. One may observe from Eq. 5.4 that $y = \hat{y} = 0$ results on zero division. Therefore, we suppose below that at least one of y and \hat{y} are different from zero.

$$J_p(y, \hat{y}) = \frac{y + \hat{y}^p - 2 \cdot y \cdot \hat{y}}{(y + \hat{y}^p - y \cdot \hat{y})} \quad (5.4)$$

$$\frac{\partial J_p}{\partial \hat{y}} = \frac{(y \cdot \hat{y})(p \cdot \hat{y}^{p-1} - y)}{((y + \hat{y}^p - y \cdot \hat{y}))^2} - \frac{y}{(y + \hat{y}^p - y \cdot \hat{y})} \quad (5.5)$$

Let us consider the case where $y = 1$ so we replace it in Eq. 5.5 and solve to find the valid values for p based on the the minimum of the derivative of the loss function.

$$\frac{\partial J_p}{\partial \hat{y}} = 0$$

$$\frac{\hat{y} \cdot (p \cdot \hat{y}^{p-1} - 1)}{(1 + \hat{y}^p - \hat{y})^2} - \frac{1}{(1 + \hat{y}^p - \hat{y})} = 0$$

$$p \cdot \hat{y}^p - \hat{y} = 1 + \hat{y}^p - \hat{y} \quad (5.6)$$

If $p = 1$ there is not minimum as shows Figure 5.3. But, if $p > 1$

$$\hat{y} = \sqrt[p]{\frac{1}{(p-1)}}$$

Note that the minimum of the loss function must not be below $y < 1$. It will lead to converge to a non desired value during optimization.

$$1 \leq \frac{1}{(p-1)}$$

$$p \leq 2 \quad (5.7)$$

If $p = 2$, the minimum of Eq. 5.6 will be exactly at $\hat{y} = 1$. If $1 < p \leq 2$, the minimum of J_p beyond 1 which is not a problem as by construction \hat{y} cannot be larger than 1. If $p > 2$, then the minimum will be between 0 and 1. Finally, if $p \leq 1$ there is no minimum. In Figure 5.4, we illustrate the minimum value of the power Jaccard according to the variation of p term. The blue line is the minimum of the loss function; red line is the ground truth value $y = 1$; and the green line is $p = 1$ where the minimum is not defined.

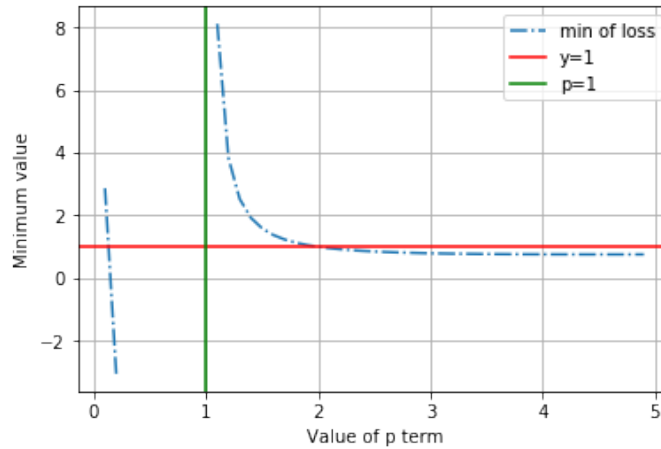


Figure 5.4: Variation of minimum of power Jaccard loss according to p value.

5.6 Experimental design

We have evaluated our proposal with several segmentation datasets to demonstrate that it can be extended to different types of images and tasks. In this section we present experimental design using Jaccard loss with three types of 2D data: 1) Grayscale images, 2) RGB Images; 2) Point clouds projections. We selected U-net based architectures and performed some variations to the model (number of filters), the training stage (dataset size and batch size), and compared the incidence of our proposal. Each configuration was repeated several times to evaluate stability and repetitiveness, which is a common issue in neural networks [159]. As evaluation metrics, we used mean IoU, accuracy, and recall scores.

5.6.1 Grayscale images

We evaluated our proposal using grayscale images with different types of experimnts. In all cases, we used the MNIST dataset for segmentation, introduced in Section 2.3. Two segmentation problems were tackled: 1) Binary segmentation to distinguish between background and digit pixels; 2) Multiclass segmentation in ten classes. In both cases, we selected a U-net model with three levels of depth where the number of filters was changed between two, four, and eight. Diverse batch sizes were used: 1, 10, and 50. Each configuration was evaluated with different losses: binary or categorical CE, classical Jaccard, and power Jaccard with several values of p . In all trained models, the input shape is a single-channel image. Each experiment was repeated five times. The code is available at ¹.

¹The code to train a segmentation model on MNIST images varying the loss functions is available at <https://github.com/daduquea/powerLosses/>

5.6.2 Color images

Three color datasets are used: 1) Aerial images from [6] for binary segmentation; 2) Plant phenotyping images from CVPPP dataset [5] for binary segmentation; 3) Urban scenes images from Cityscapes [4] for multiclass segmentation. These datasets are introduced with more detail in Section 2.3.

Aerial images

We performed two binary segmentation tasks with aerial images: 1) Road vs no-road; 2) Building and no-building. In both tasks, we used data splits provided by the authors as follows: 1108 images for training, 14 for validation, and 49 for testing for road detection; 137 images for training, four for validation, and ten for testing, for building detection.

We applied transfer learning and selected a U-net model initialized with weights trained on ImageNet. We chose Adam optimizer with a default learning rate of 10^{-3} and a patience equal to five. We only changed the loss function in different experiments by: crossentropy, focal loss, Dice, Jaccard, and power Jaccard with $p = 1.5$ and $p = 2$.

Urban scene images

Cityscapes dataset is composed of 5K color images divided into three subsets: training (2975), validation (500), and test (1525) and annotated with 30 classes in the context of autonomous driving. In Section 2.3 is presented a more detailed description of the dataset and some examples of the type of images that it contains.

We selected a subset of four classes relevant for autonomous driving to perform semantic segmentation in unbalanced data: person, car, road, and background. We trained a U-net architecture and compared the following loss functions: categorical cross-entropy, classical Jaccard, power Jaccard with $p = 1.5$, and power Jaccard with $p = 2.0$. All the other parameters during training were fixed except the loss function.

Plants phenotyping images

CVPPP dataset is commonly used to study plant growing and leaf counting. In our case, we performed binary segmentation tasks in order to identify *plant* vs *background* pixels. We selected the train and validation data splits from A1 subset, composed of 106 and 22 images, respectively. For the testing set, we selected A2 subset composed of 31 images.

We selected this dataset to evaluate the performance of several loss functions compared to power Jaccard loss. We selected a classical U-net architecture with 16 filters on the first layer. We followed the same training protocol for all experiments: Adam optimizer with a default learning rate of 10^{-3} and a patience equal to five. The unique variation between different experiments was the loss function.

5.6.3 Point clouds

We evaluated power Jaccard using 3D point clouds using two different data representations: first, with BEV projections (this section); second, working directly in 3D raw data (Section 6.3.2). In both scenarios, we evidenced that our proposal improves performances compared to classical loss functions.

Projections of SHREC data

Street 3D [10] dataset contains 80 point clouds, each one has about three millions points (x, y, z) . Each point cloud of the training set is manually labeled with the following classes: *building*, *car*, *ground*, *pole* and *vegetation*. Segmentation is obtained from 2D Bird eye view (BEV). Figure 5.5 shows a point cloud with ground truth labels.

We divided segmentation of 3D point clouds in two simpler problems: **1)** Segment *lower* points (low slice) in *building* and *car* classes; **2)** Segment *higher* points (high slice) in *building* and *vegetation* classes. Figure 5.5(a) shows BEV projections of *low* and *high slice* Figure 5.5(b).

Ground and *poles* have been discarded because: **1)** *Ground* can be extracted by λ Flat Zone method proposed by [160] to compute Digital Elevation Model (DEM). This is an analytical approach that allows to detect ground-like classes; **2)** *Pole* class is problematic because a single traffic sign, very different from other pole instances, contains 70 % of the *pole* class points in the whole dataset.

We computed hand crafted features based on the BEV projections [54] from 3D point clouds. For the low slice: h_{max} and $max(h_{max}, \Delta h_{min})$ and for the high slice: h_{min} , h_{max} and Δh . We note that h_{max} and h_{min} represent the maximum height and the minimum height of all points that fell in the same pixel, $\Delta h = h_{max} - h_{min}$ and $\Delta h_{min} = h_{min} - DEM$.

The semantic segmentation task was performed independently on each slice: one model was trained in the low slice and another one in the high slice. Both slices share some common characteristics such as the same U-net-based architecture with three levels of depth, the kernel size of convolutions, patience and Adadelta optimizer [161] with a learning rate of 0.001

5.7 Results

In this section, we present obtained results in binary and multiclass segmentation tasks. We demonstrate that our proposal performs better than classical losses in evaluated scenarios, including different types of images and point clouds (see Section 5.6.3).

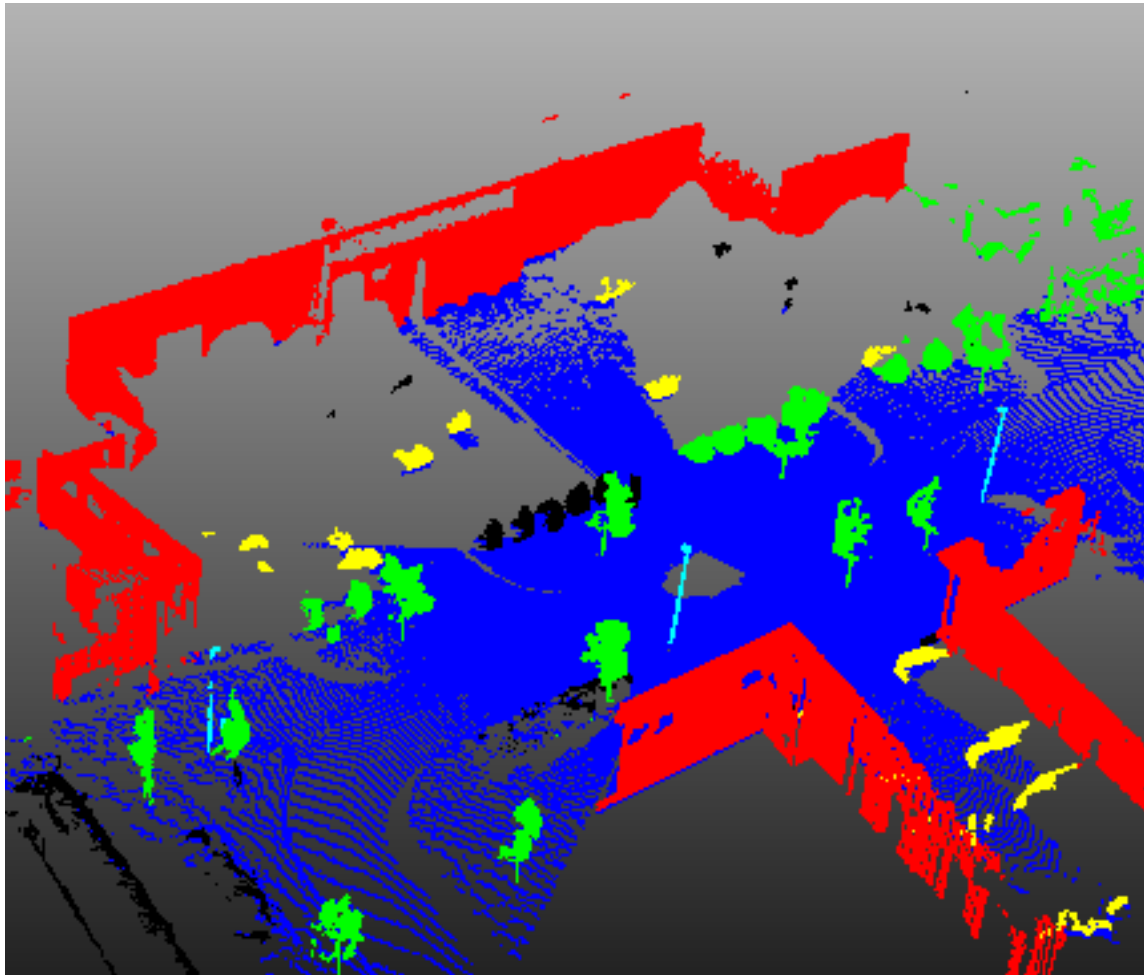
5.7.1 Grayscale images

Binary segmentation

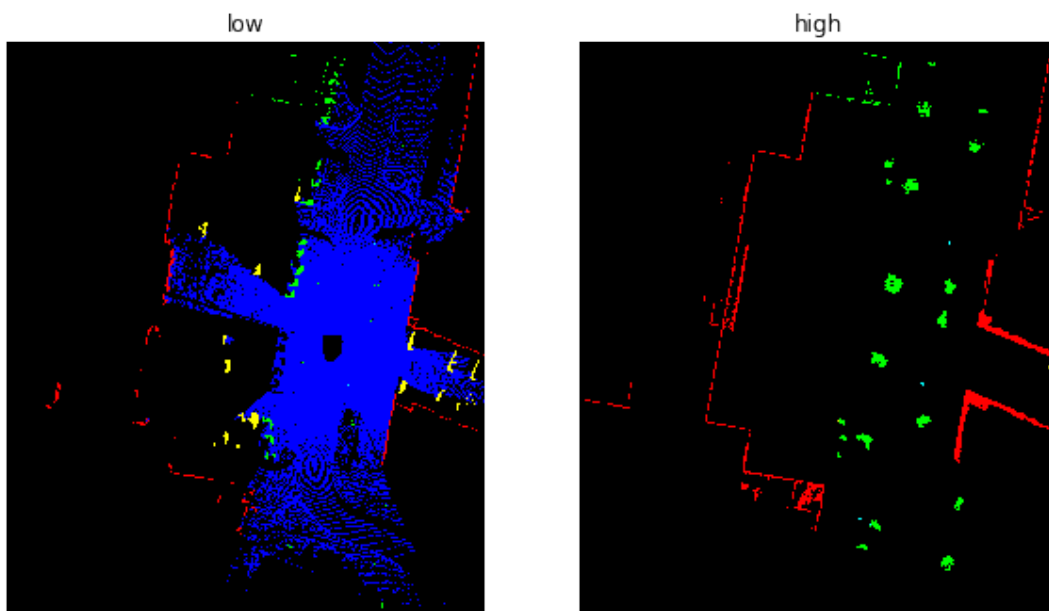
Experiments with a different number of filters, batch size, and loss functions are performed. Table 5.1 shows the results using a batch size equal to one. Mean IoU, standard deviation, and the best IoU of five runs are reported.

We experimentally found that increasing the batch size negatively affects the performance of the model. It can be justified because, with a smaller batch, the model gradually learns to distinguish between background and a single class. Even though, over a batch size of 10, the variance of the *digit* class increases because it groups a set on non-homogeneous instances of the ten classes. Furthermore, Table 5.1 reports a high standard deviation for almost all configurations. It implies that during training, models converged to different local minima with different values at each run.

Power losses privilege to train simpler models outperforming crossentropy and classical Jaccard. Table 5.1 shows that the model with two filters and power Jaccard with $p = 1.75$ obtained a performance equal to the best model with eight filters. The improvement obtained using power losses is higher when training smaller models. These loss functions could be helpful with low memory requirements.



(a) 3D points



(b) Bird Eye View (BEV) projections

Figure 5.5: Point cloud with ground truth from Street 3D [10] dataset and the corresponding BEV projections.

Filters	Metric	CE	Jac. $p = 1$	Jac. $p = 1.25$
2	IoU	0.8542 ± 0.1651	0.4402 ± 0.0298	0.5303 ± 0.2101
	Best IOU	0.9884	0.5000	0.9507
4	IoU	0.8773 ± 0.1889	0.4551 ± 0.0366	0.5545 ± 0.2234
	Best IOU	0.9878	0.5000	0.9977
8	IoU	0.8216 ± 0.1654	0.4403 ± 0.0304	0.4348 ± 0.0191
	Best IOU	0.9504	0.5011	0.4737
Filters	Metric	Jac. $p = 1.5$	Jac. $p = 1.75$	Jac. $p = 2$
2	IoU	0.5538 ± 0.2217	0.9813 ± 0.0172	0.7679 ± 0.2797
	Best IOU	0.9735	0.9977	0.9975
4	IoU	0.5396 ± 0.2290	0.6507 ± 0.2760	0.7687 ± 0.2804
	Best IOU	0.9566	0.9938	0.9977
8	IoU	0.5395 ± 0.2290	0.5397 ± 0.2289	0.5397 ± 0.2289
	Best IOU	0.9977	0.9977	0.9977

Table 5.1: Binary segmentation in MNIST dataset with batch size of one. In all cases, U-net architecture with three levels of depth and varying the number of filters in the first layer.

Batch	Metric	CE	Jac. $p = 1$	Jac. $p = 1.25$
1	IoU	0.0956 ± 0.0309	0.0000 ± 0.0000	0.0062 ± 0.0076
	Best IOU	0.1501	0.0000	0.0152
10	IoU	0.1341 ± 0.0228	0.4537 ± 0.1014	0.5298 ± 0.0963
	Best IOU	0.1612	0.6562	0.6378
50	IoU	0.1534 ± 0.0101	0.4819 ± 0.0779	0.5255 ± 0.1055
	Best IOU	0.1679	0.6188	0.7342
Batch	Metric	Jac. $p = 1.5$	Jac. $p = 1.75$	Jac. $p = 2$
1	IoU	0.4831 ± 0.4031	0.6388 ± 0.3394	0.5160 ± 0.4240
	Best IOU	0.9137	0.8960	0.9450
10	IoU	0.6852 ± 0.0464	0.8307 ± 0.0535	0.7953 ± 0.0499
	Best IOU	0.7747	0.8793	0.8856
50	IoU	0.6364 ± 0.0699	0.7403 ± 0.1137	0.7590 ± 0.0577
	Best IOU	0.7432	0.8342	0.8064

Table 5.2: Mean IoU in multiclass segmentation on MNIST.

Multiclass segmentation

We varied batch size between 1, 10, and 50 and repeated each configuration five times, as presented in Table 5.2.

From Table 5.2, it can be seen that power Jaccard allows obtaining a higher score compared against cross-entropy and classical Jaccard. In all conducted experiments, the performance of our proposal was equal or better than classical losses using grayscale images.

Figure 5.6 presents some predictions obtained with the best model achieved using power Jaccard with $p = 2$ and batch size of one. We observed that in the best models, the errors were in general at a pixel-wise level. This type of error can be solved through regularization techniques such as voting systems [162].

5.7.2 RGB images

Aerial images

Table 5.3 presents accuracy values by experiment with several loss functions. We observe that in both experiments, the use of power losses leads to better results by a small margin, and the highest accuracy was obtained with $p = 1.5$. One may also note that BCE has the lowest validation

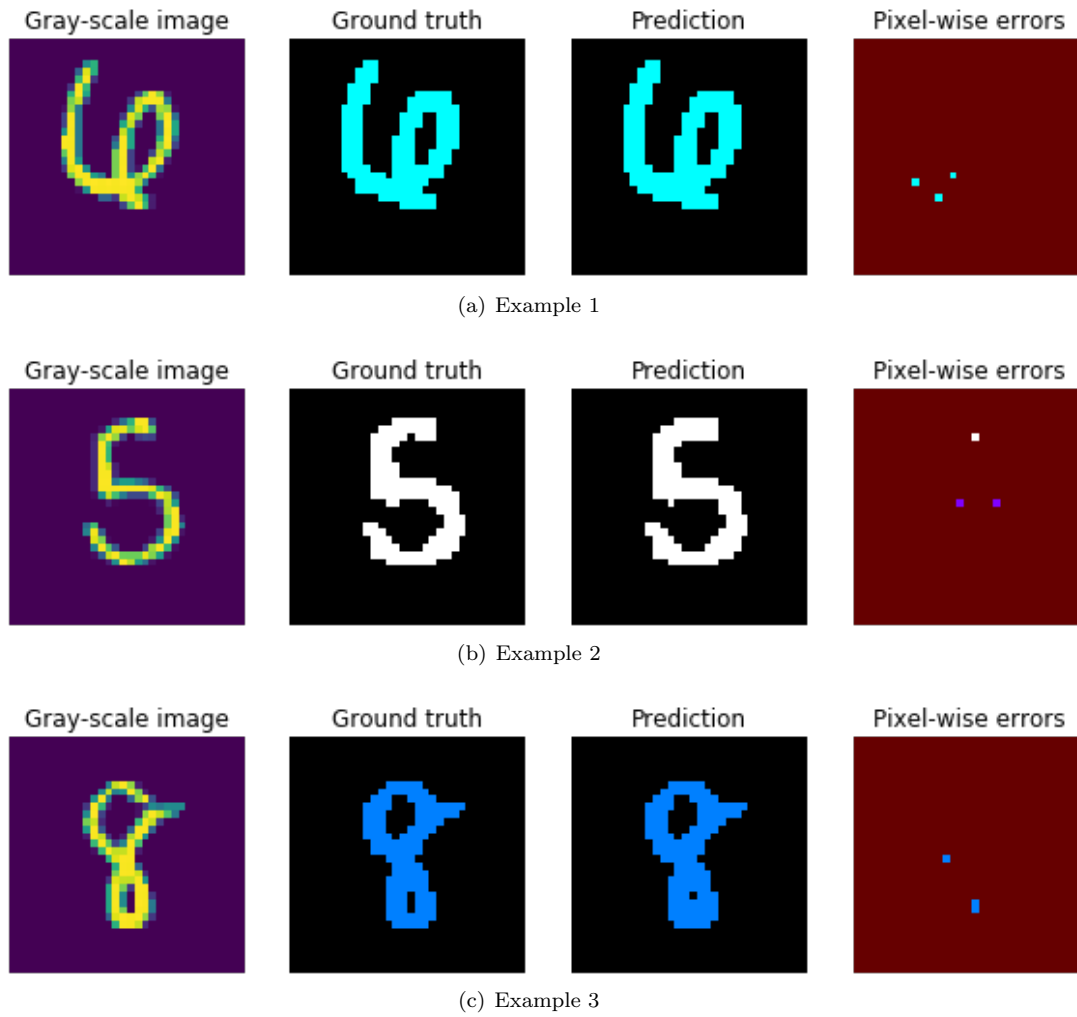


Figure 5.6: Multiclass segmentation using MNIST dataset. From left to right: 1) Original gray scale image; 2) Ground truth generated from gray scale image; 3) Prediction using best obtained model; 4) Errors between prediction and ground truth.

	Roads	Buildings
BCE	0.9622	0.9376
F-BCE	0.9677	0.9421
Dice	0.9680	0.9489
Jac. p = 1	0.9676	0.9484
Jac. p = 1.5	0.9684	0.9503
Jac. p = 2	0.9682	0.9502

Table 5.3: Accuracy in validation set on Aerial images. First column indicates the used loss function: BCE, Focal BCE, Dice score and power Jaccard.

	CE	Jac. p = 1	Jac. p = 1.5	Jac. p = 2
Person	0.1135	0.0000	0.1118	0.1390
Car	0.4620	0.4004	0.4209	0.5082
Road	0.8380	0.8137	0.8047	0.8263
Background	0.8721	0.8541	0.8605	0.8738
Mean IoU	0.5714	0.5170	0.5495	0.5868

Table 5.4: IoU by class and mean IoU in validation set of Cityscapes [4].

accuracy in both datasets. The small differences between different loss functions occur because of an already trained model as a feature extractor, as described later in this chapter.

Urban scene images

Table 5.4 reports the results on Cityscapes dataset. Power losses improve the performance on less populated classes such as person and car thanks to the higher penalty for worst predictions (see Figure 5.3).

The *relative improvement* in IoU between cross-entropy and power Jaccard with p equal to 2 in *background* was 0.1949%, in *person* was 22.46% and in *car* was 10%. On the *road*, it worsened by 1.39%. In the *mean IoU*, the relative improvement was 2.69%. Figure 5.7 shows the predictions of the image presented in Figure 2.5 using the models trained with different losses. It is seen how the influence of the power term qualitatively improve the segmentation of the person class.

Plants phenotyping dataset [5]

In Table 5.5, we report obtained results on binary segmentation with different loss functions. We compared: binary crossentropy (BCE), Focal BCE (F-BCE), Dice, Tversky with $\alpha = 0.1$ (TV1) and $\alpha = 0.5$ (TV5), classical Jaccard and power Jaccard with $p = 1.2, 1.4, 1.6, 1.8, 2.0$.

From obtained results, we demonstrate that the use of power Jaccard improves performance also in this task compared to classical losses. We identified that the best performance was obtained with $p = 1.4$. It is a relevant finding because it illustrates that a precise selection of p value may improve the model.

5.7.3 Point clouds projections

This section presents the results of semantic segmentation in point clouds projections. We divided each point cloud into two slices: low and high, to simplify classification problems and focus on the impact of loss function's selection. Tables 5.6 and 5.7 present obtained results with several losses.

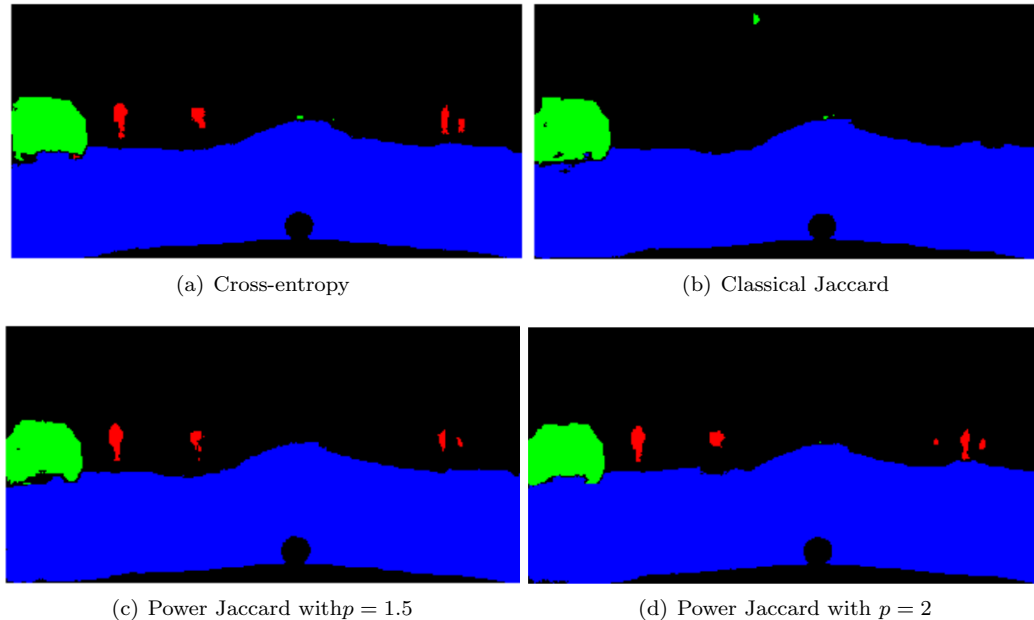


Figure 5.7: Example of Prediction on Figure 2.5 in Cityscapes. Road (blue), person (red), car (green) and background (black). Note that Power Jaccard performs better on smaller classes than the classical one. Quantitative results are given in Table 5.4.

We report three values by loss function: IoU is the average and the standard deviation of the mean IoU in the test set and $best IoU$ is the highest IoU obtained in test set

5.8 Discussion

We presented obtained results on 2D datasets with proposed Power Jaccard loss for segmentation tasks. In tested scenarios, it was observed that the use of these loss functions improves the performance and the rate of convergence, as previously presented with point clouds segmentation in Section 5.5. One may observe that with our proposal, the mean IoU is higher than with classical losses and the standard deviation is smaller when the model converges.

The experiments carried out with the MNIST dataset were grouped into two different segmentation tasks: binary (digit and background) and multiclass (ten digits). In both cases, we evaluated the influence of the loss function during training. In the first task, we changed the number of filters of the model between 2, 4, and 8. Table 5.1 presents obtained results. One may note that with the simplest architecture with 2 filters, most of the models converge and obtain a good performance above 95%. Also, it was observed that the highest IoU was obtained with power Jaccard loss and $p = 1.75$. In the case of 4 and 8 filters, the best performance was obtained using $p = 2$ by a small margin. We highlight that in the case of Power Jaccard with $p = 1$, equivalent to classical Jaccard, no models converged.

The second task with the MNIST dataset was carried out to evaluate the influence of loss

	BCE	F-BCE.	Dice	TV1	TV5	Jac.
Plant	0.696	0.621	0.709	0.695	0.730	0.619
Background	0.993	0.991	0.992	0.991	0.992	0.991
Mean IoU	<u>0.844</u>	<u>0.806</u>	<u>0.850</u>	<u>0.843</u>	<u>0.861</u>	<u>0.805</u>
	Jac. p = 1.2	Jac. p = 1.4	Jac. p = 1.6	Jac. p = 8	Jac. p = 2	
Plant	0.735	0.778	0.719	0.652	0.619	
Background	0.993	0.994	0.993	0.992	0.991	
Mean IoU	<u>0.864</u>	0.886	<u>0.856</u>	<u>0.822</u>	<u>0.805</u>	

Table 5.5: Performance in test set of CVPPP dataset [5] with different loss functions. Each column corresponds to different loss function as follows: binary crossentropy (BCE), Focal BCE (F-BCE), Dice, Tversky with $\alpha = 0.1$ (TV1) and $\alpha = 0.5$ (TV5), classical Jaccard and power Jaccard with $p = 1.2, 1.4, 1.6, 1.8, 2.0$.

Metric	CE	Focal BCE	Dice score
IoU	0.279 \pm 0.426	0.450 \pm 0.172	0.665 \pm 0.196
Best IOU	0.934	0.797	0.798
Metric	Jac. p = 1	Jac. p = 1.5	Jac. p = 2
IoU	0.702 \pm 0.230	0.931 \pm 0.009	0.925 \pm 0.009
Best IOU	0.941	0.943	0.939

Table 5.6: Performance in low slice from SHREC'20 dataset with different loss functions.

Metric	CE	Focal BCE	Dice p = 1
IoU	0.341 \pm 0.000	0.570 \pm 0.016	0.427 \pm 0.173
Best IOU	0.341	0.605	0.787
Metric	Jac. p = 1	Jac. p = 1.5	Jac. p = 2
IoU	0.341 \pm 0.000	0.761 \pm 0.144	0.761 \pm 0.020
Best IOU	0.341	0.809	0.788

Table 5.7: Performance in high slice from SHREC'20 dataset with different loss functions.

function in a more complicated segmentation problem. In this case, we changed batch size during training between 1, 10, and 50. Table 5.2 presents obtained results in this task. It was found that the best results were obtained with power Jaccard for all batch sizes. One of the main findings in this set of experiments was that small batches help the model during training because it learns how to distinguish between background and a single class at each time. This can be useful in those cases where the classes group a set of non-homogeneous instances.

We tested the proposed power Jaccard with different datasets of RGB images: aerial images, Cityscapes, and CVPPP. We carried out binary and multiclass segmentation to validate our proposal. We performed two binary segmentation with aerial images, and the best results were obtained with $p = 1.5$ in both cases. With Cityscapes, multiclass segmentation in four classes was performed. In this case, minority classes were improved with $p = 1.5$. With CVPPP images, we performed binary segmentation, and the best results were obtained with $p = 1.4$.

As the last set of experiments, we evaluated power Jaccard with point cloud projections from the dataset proposed in SHREC'20 challenge [10]. Experiments were carried out later than during the competition described with more detail in Section 6.3.2.

We tested our proposal with projection-type images to demonstrate that our proposal could be generalized for other types of data, apart from grayscale and RGB images. With point clouds projections, we performed binary and multiclass segmentation tasks. The use of *Power Jaccard* improved the performance and rate of convergence of models (more models converged using same initialization parameters). As it is presented in Table 5.6, standard deviation decreases when p value increases. Additionally, we observed CE and Focal BCE do not converge as often as the proposed losses in tested scenarios. Even though, their best IoU in Table 5.6 is comparable with power functions. Models trained with $p = 1.5$ perform better than classical Jaccard in both slices.

One of the most relevant findings of power losses was that they permit to train simpler models than classical losses. From Table 5.1, it is seen that the model with two filters and power Jaccard with $p = 1.75$ performs as well as the best model with eight filters. This advantage can be helpful for applications with memory constraints such as cellphones, tablets, or even low-cost computers.

5.9 Conclusions

In this chapter, we present our proposed power Jaccard loss for semantic segmentation tasks with 2D images. We carried out different experiments including grayscale and color images and BEV projections of point clouds. Obtained results demonstrate that our proposal improves performance and rate of convergence in different scenarios. We also show that simpler models can be trained and still obtain comparable results if power Jaccard is used. For applications with memory constraints, the use of power Jaccard is suggested over other classical losses.

Due to significant and interesting results with 2D images, we also studied the influence of power Jaccard directly on 3D point clouds. In Chapters 6, we evaluated the influence of the loss function in different scenarios using our proposed Paris Carla 3D dataset. Similar to the results presented with images, power Jaccard performs better than classical loss functions.

In the same spirit of the new loss function presented in this chapter, we proposed a hierarchical loss function to improve semantic segmentation in imbalanced datasets. It is introduced in Section 6.4. This loss function permits embedding some *a-priori* knowledge about the relationship between classes and improving the performance of the models.

Chapter 6

Segmentation of point clouds

6.1 Résumé

Dans ce chapitre, nous présentons nos contributions à la segmentation sémantique et d’instances des nuages de points. Nous présentons une revue d’architectures récentes d’apprentissage profond pour la segmentation sémantique. Ensuite, nous évaluons différentes alternatives pour effectuer cette tâche en utilisant des projections 2D et un traitement directement 3D des données. Nous proposons une approche pour intégrer des connaissances expertes au cours de l’apprentissage en utilisant une représentation hiérarchique des classes. Dans la dernière section du chapitre, nous étudions l’influence de la couleur dans la segmentation sémantique des nuages de points.

6.2 Introduction

In this chapter, we introduce proposed approaches to perform segmentation tasks in urban point clouds. We studied two related applications: semantic segmentation and instance segmentation. Performed experiments were carried out with datasets in the context of autonomous driving such as Semantic KITTI [9], Street 3D [10] and the novel Paris Carla 3D.

The content of this chapter is divided as follows: Section 6.3 introduces state of the art in semantic segmentation for point clouds; Section 6.3.2 presents proposed approaches using 2D projections and directly on 3D for semantic segmentation tasks; Section 6.4 raises proposed methods to handle data imbalance; Section 6.5 presents instance segmentation; Section 6.7 displays discussion and result analysis; Section 6.6 presents an analysis about of the influence of RGB features in urban scene segmentation; finally, conclusions are presented in Section 6.8.

The main contributions of this chapter are: participation in SHREC challenge with our proposed Spherical DZNet [10]; the novel Paris Carla 3D dataset; a hierarchical loss for semantic segmentation tasks; and a study of color influence in urban scene segmentation.

6.3 Semantic segmentation

3D Semantic segmentation is the process of partitioning a point cloud into multiple homogeneous regions [29]. According to the application, the criteria to group points into regions may vary. For

example: in part segmentation tasks, points representing the same part of the class have the same label; in semantic segmentation tasks, points belonging to the same object will have the same label. In this work, we studied different approaches to perform semantic segmentation of urban point clouds by using supervised and unsupervised methods.

Segmentation of point clouds is a very active research topic with increasing interest over the last few years. Figure 6.1 displays the number of works published on Google Scholar by searching “urban 3d point cloud segmentation” between 2000 and 2020. As we presented in Chapter 2, the availability and expansion of datasets have motivated the scientific community to tackle automatic tasks in point clouds using different approaches.

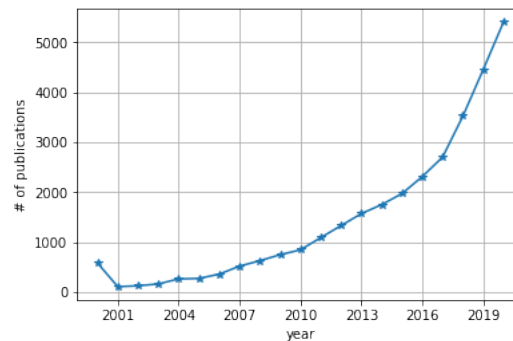


Figure 6.1: Number of publications by year in Google Scholar with the query “urban 3d point cloud segmentation”

6.3.1 Related Work

This section introduces some of the most relevant works in semantic segmentation of 3D point clouds for urban scenes. We highlight that it is not a complete revision of all existing methods but a selection of the most relevant ones related to our research. The interested reader may refer to [163] and [65] for a detailed description of recent methods.

Over the last years, many methods have been proposed to perform semantic segmentation of point clouds. The number of different strategies is associated not only with the difficulty of the task but also with some characteristics of point clouds such as lack of grid structure, permutation invariance, sparsity, density variation, among others. Additionally, the size of dense urban point clouds limits the implementation of some methods due to their enormous computational cost.

Because of the wide range of existing methods, the way they can be categorized has also evolved. Initially, methods were based on hand-crafted features and classical classifiers. According to [29], most of the methods before 2014 can be classified according to the criteria to characterize objects: edge-based, region-based, attributes-based, model-based, and graph-based.

For example, edge-based methods describe objects according to the shape of a contour. The working principle of some algorithms such as [164] and [165] is to detect points with high gradients in intensity or unit normals. In region-based methods, *similar* points are grouped by analyzing their neighborhoods, either using seeds (bottom-up methods) or dividing regions (top-down methods). A detailed survey about methods to segment point clouds by using classical approaches can be found

in [29].

After the boom of deep learning for semantic segmentation in 2D images and the availability of point clouds datasets, many new methods have been proposed. According to [52] and [65], methods can be also classified according to the way they represent 3D data. In this case, most of the methods can be grouped in: 1) 2D and 3D structured grids; 2) Directly on 3D point cloud.

The first group is composed of methods that rely on representations of point clouds on structured grids. They allow dealing with some common issues of 3D point clouds such as data sparsity, density variation, and orderless. The representation on structured grids can be subdivided into two categories: 2D projections and 3D volumetric representation.

The representation as 2D projections allows us to take advantage of the maturity of existing machine learning algorithms with images and apply them to perform tasks with 3D point clouds. Some of the most relevant methods that rely on 2D projections are listed below.

First methods were based on hand-crafted features such as [160, 166] and [167], used top view projection and *statistical* classification algorithms. Then, due to the boom of deep learning in images and its incursion on 3D point clouds, many other methods surged such as MultiviewCNN [168], RotationNet [169] and stacked local convolutional autoencoder (SLCAE) [170].

The second type of structured representation is based on 3D grids. In this case, point clouds are voxelized to perform 3D convolutions directly. Some known methods are: 3DShapeNets [171] that represent point clouds as probabilities distribution of binary variables in a 3D grid; VoxNet [172] also builds probability distributions and includes a fully connected layer as the last layer. The main issue of structured representations is associated with high memory consumption directly dependent on voxel size. Point cloud resolution is affected, and small details and objects can be lost.

The second group works directly on 3D point clouds, and the first published work using this approach was PointNet [17]. This model extracts global features of points, but local information about neighborhoods is lost. To overcome with this issue, PointNet++ [18] was proposed to extract local features based on regions. Other works have been proposed seeking to exploit local information of points such as KPConv [19], PointCNN [66] and SparseConv3D [173]. A recent survey with a more detailed explanation of the state of the art DL models can be found at [65] and [52].

DL architectures used in this thesis

In this chapter, we worked with architectures to perform semantic segmentation tasks. Firstly, we worked with U-net architecture [138], commonly used with images, but in our case, we used it with 2D projections of point clouds (see Section 6.3.2). Then, we worked directly with 3D data using PointNet [17], PointNet++ [18] and KPConv [19] models. A description of the main characteristics and advantages/disadvantages of selected architectures is presented below.

PointNet [17] PointNet was one of the pioneer models working directly on 3D data. This architecture was developed to perform object classification, part segmentation, and semantic segmentation. The main characteristic of PointNet is to be robust to input perturbations and corruptions. According to the authors, the architecture was designed to deal with orderless, transformation invariance, and neighborhood interaction as follows:

1. Orderless: Point cloud data is not sorted by nature and it was required to build a model capable of processing data and obtaining the same results even if the order of points changes.

2. Transformation invariance: Objects can be located in any place on the scene without priors. The model must obtain the same output for objects placed at different spatial coordinates and orientations.
3. Neighborhood interaction: Points can not be solely processed due to their natural interaction with neighbors that provide local features.

Figure 6.2 displays PointNet architecture composed of the following modules:

- Spatial transformers [174]: Point cloud is aligned to canonical space by applying a rigid or affine transformation. They are located before each Multilayer Perceptron (MLP). The first transformer takes input coordinates and predicts the affine transformation 3×3 matrix. The second transformer is located before feature extraction and learns the 64×64 matrix to align features from different point clouds. In the second case, a regularization term is included to help convergence due to higher dimensions of the matrix.
- Multilayer Perceptron (MLP): Shared MLPs to extract features by point are applied. The first maps from 3 dimensions (x, y, z) to 64 dimensions by point. The second maps from 64 to 1024 dimensions. As the last layer of the model, an MLP is used to predict output scores of the network.
- Local and global information combination: Max-pooling is used as symmetric operator to make PointNet invariant to input permutation. The concatenation of local and global features also permits the use of the network for segmentation tasks.

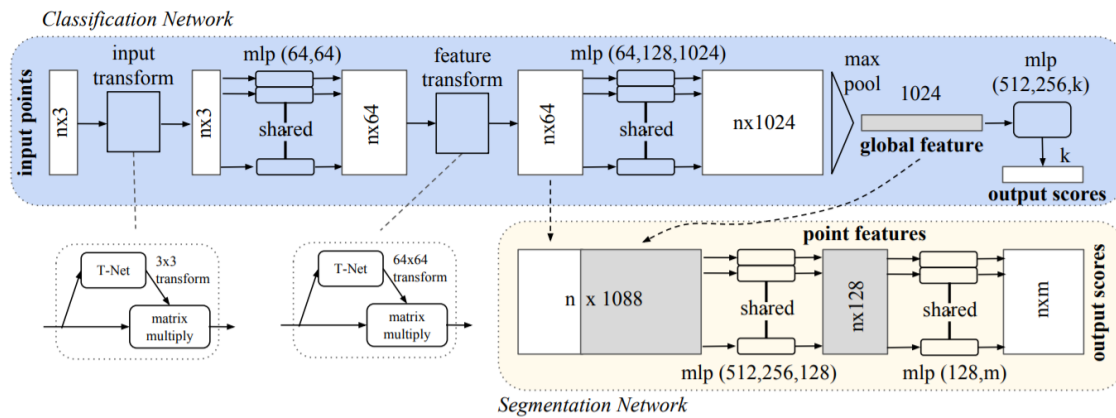


Figure 6.2: PointNet architecture proposed by [17]

From Figure 6.2, it can be observed that PointNet architecture is simple and intuitive. It maps each point n from 3 to 64 dimensions and then from 64 to 1024 dimensions. Next, using the max-pooling operator, a vector including global features of 1024 dimensions is obtained. The last section of the network changes according to the type of task of the network: for classification tasks, MLP is used to map global features from 1024 dimensions to k output scores; for segmentation tasks,

local features (64 dimensions) and global features (1024 dimensions) are concatenated to obtain a 1088 dimensions vector per point. Then, by using MLP, the features vector is mapped to 128 and again to k . The output of the segmentation model is an array of $n \times m$ dimensions.

When the PointNet model was proposed, it was the state of the art in most point cloud segmentation and classification competitions. It was the seminal work that strongly influenced the use of 3D point clouds instead of grid-based approaches. Even though, some significant limitations of this model have been identified such as:

1. Lack of local spatial relationships between points. Extracted features represent most of the global characteristics of point clouds without local context.
2. Global features depend on absolute coordinates due to spatial transformers. It makes it harder to predict new point clouds with unseen configurations.
3. Elevated computational cost that limits the size of input point clouds.

PointNet++ [18] PointNet++ was proposed as an extension of the initial PointNet architecture to tackle some of the issues identified in the model. This new architecture captures local information in point clouds employing the basic CNN structures where its low-level neurons with small receptive fields can abstract local patterns. Moreover, a hierarchical network composed of several PointNet architectures can learn new features by combining different scales and point densities. Figure 6.3 displays PointNet++ architecture.

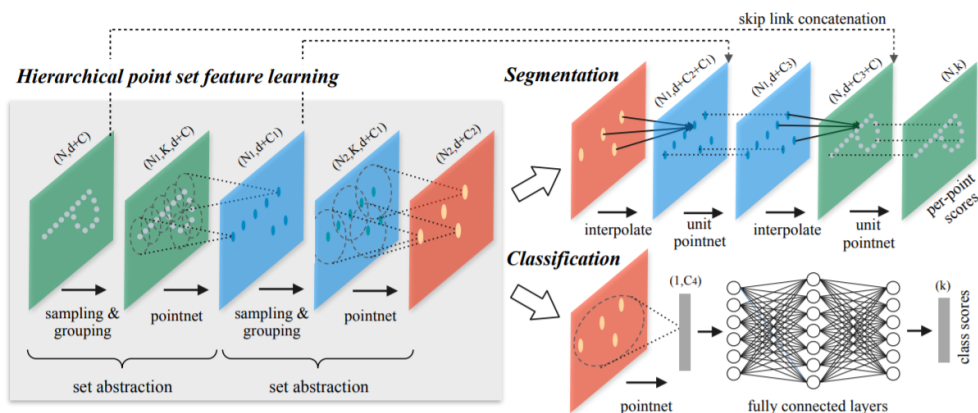


Figure 6.3: PointNet++ architecture proposed by [18]

The main difference between PointNet and PointNet++ is that the first uses a single max-pooling operator to aggregate the whole point set while the second builds a hierarchical grouping of points with progressively larger local regions along the hierarchy. The architecture is composed of a set of abstraction layers, where each level performs the following three layers:

1. *Sampling layer*: From a set of N points, sample N_1 points. It is achieved with Farthest Point Sampling (FPS), presented in Section 2.4.

2. *Grouping layer*: For each sampled point, find its neighborhood using local coordinates. It is performed with radius query search instead of KNN. It guarantees a fixed region scale.
3. *PointNet layer*: Apply a shared mini-PointNet to each neighborhood to extract local features. It encodes the centroid's neighborhood.

A significant improvement of this architecture is its capability to learn under non-uniform sampling density. Authors provide two alternatives, according to the type of task of the network and point clouds size: 1) Multiscale Grouping (MSG): Learn features at different scales with PointNet network and then concatenate it to form a multiscale feature. It is computationally expensive because features are extracted for large scale neighborhoods and every centroid point; 2) Multiresolution grouping (MRG): This strategy groups two vectors, one from the features extracted on a lower abstraction level (L_{i-1}) and the other obtained by directly processing all raw points, using a single point net. This approach gives more relevance to higher-density regions than PointNet model. If the density of the local region is high, the first vector is more reliable; otherwise, the second vector should be weighted higher.

The last part of the network is composed of fully connected layers that compute a k vector with output scores for classification tasks. For semantic segmentation tasks, the last part of the architecture is slightly different because must provide point-wise features to all input points and not only to subsampled ones. It is carried out by using distance-based interpolation to propagate learned point features from different abstraction levels. This process is repeated until every point from input data has all propagated features.

This architecture achieved state-of-the-art results in most of the classification and segmentation tasks when it was published. It demonstrated a significant improvement compared to the initial PointNet model.

KPConv [19] This model introduces a new design of point convolution name Kernel Point Convolution (KPConv) that operates directly in 3D data. One of their main characteristics is the capacity to deal with varying densities of point clouds thanks to a regular subsampling strategy based on multiscale spherical neighborhoods [71]. Then, by using Kernel Point Network layers, the proposed convolution and subsampling strategy can be integrated into a deep learning architecture.

A KPConv layer calculates new features for a given point based on its location, current features, and neighborhood. Similarly, as it occurs with CNN in 2D images, these new features are the sum of all the kernel values multiplied by the neighbor's features. In Figure 6.4 is illustrated a simple comparison between 2D convolutions using CNN in images and 3D convolution using KPConv.

The proposed subsampling strategy is based on a single parameter ρ that controls points density. It allows to have regularly positioned points in each neighborhood and, consequently, regular kernel points. These kernel points must be located as far as possible from center point x but also be contained in its neighborhood. The authors of the architecture proposed to use multiple kernel points to capture different features in 3D data.

A very important concept of KPConv is the kernel function. It takes a neighbor position x_i and centers it on the given point x where the convolution is performed. It is calculated by:

$$y_i = x_i - x \tag{6.1}$$

Based on the distance to the center point x , the weight of each kernel point varies and their influence in the calculation of the new features. This is computed using the correlation function, which provides higher values for closer points.

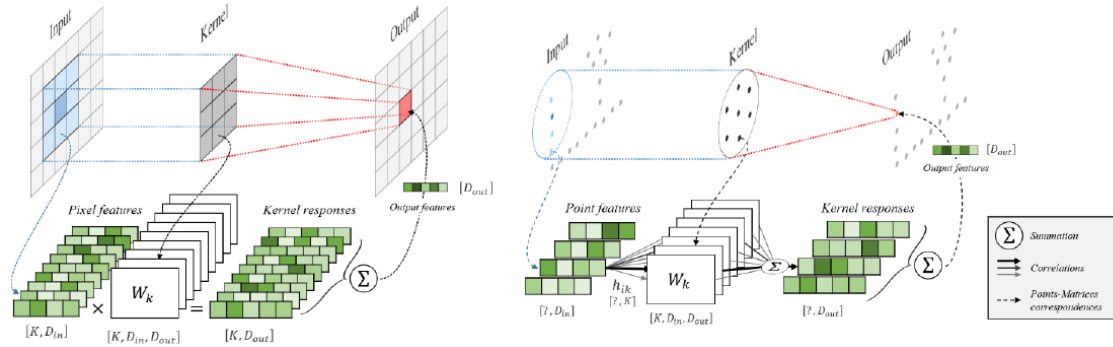


Figure 6.4: Comparison 2D convolutions and KP Convolutions. Image taken from [19].

KPConv has outperformed most of state of the art in classification and semantic segmentation tasks. By using this type of approach, three of the most common issues in point clouds are tackled:

1. Subsampling strategy based on multiscale spherical neighborhoods is robust to density variations.
2. Convolutions based on spherical neighborhoods vary weights of kernel points using a correlation function that considers their distance to the center point.
3. Orderless of point clouds does not affect KP Convolutions.

6.3.2 Proposed approaches

We studied different alternatives to perform semantic segmentation. Our proposed approaches can be grouped according to the type of representation of the point clouds: 2D based with Semantic Kitti and Street 3D datasets; 3D based using PC3D dataset. In all scenarios, we identified that class imbalance is a constant issue that strongly affects the performance of different methods. To deal with this issue, in Section 6.4 we propose some solutions using a hierarchical-based approach.

2D based

We studied two approaches based on 2D projections to perform semantic segmentation of point clouds. They both rely on spherical projections, but the second one also includes an additional stage with BEV projections. We performed experiments using Semantic KITTI [9] and Street 3D [10] datasets, using only geometrical information without intensity or color features.

Spherical projections The use of spherical projections to segment point clouds has demonstrated successful results, as we presented in Section 2.4.3. As the first approach to semantic segmentation tasks, a solution based on them was proposed. It is divided into the following steps:

1. Compute spherical projections of 3D point clouds.

2. Train a DL model based on an U-net architecture.
3. Backproject images to 3D point clouds.

Compute spherical projections Spherical projections provide an image from the scanner point of view and for some classes, it is enough to classify them. To compute a spherical projection from a given point cloud requires additional information about scanner resolution used during acquisitions. According to these values, *height* and *width* of projected image can be directly computed. An advantage of spherical projections over BEV projections is that occlusions are not present because they correspond to images acquired from the scanner point of view. It is very important to pay attention to the dimension of the projected image to obtain a good representation of the original point cloud. On the one hand, if image projection is too small, several 3D points may fall in the same pixel, and the border of the objects may be fuzzy. On the other hand, if image dimensions are too big, most of the pixels will not have 3D points projected on them, and objects' shapes can be split into several connected components with holes inside them.

With the aim to better understand the relation between image dimensions and the rate of points projected into pixels where other pixels have already been projected, some experiments were carried out. We selected the first 300 point clouds from the validation set of Semantic KITTI and computed spherical projections of ground truth with different image dimensions. Then, we computed the rate of points projected into a pixel where other points have already been projected. We compared 4 image projection sizes: 64x1024, 64x2048, 128x1024, 128x2048 and results are presented in Figure 6.5.

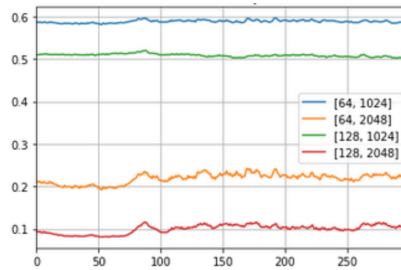


Figure 6.5: Rate of points projected in a pixel where other pixel has already been projected. Experiments performed with first 300 point clouds from validation set from Semantic KITTI dataset.

Obtained results presented in Figure 6.5 demonstrate that image dimensions are very important in spherical projections. If the image is small (64x1024), the rate of points projected into non-empty pixels is about 60 %. However, if the image is big (128x2048), it is about 10 %. It is essential to be careful with these results because the lowest rate does not necessarily mean the best: if image dimensions are too large, objects may have holes inside, and regular 2D convolutions can be affected.

Learning in 2D U-net architecture is commonly used for image segmentation tasks. Some works published between 2018 and 2019, such as SqueezeSeg [55], SalsaNet [56] and RangeNet++ [57] demonstrates that skip-connection modules work well also in 2D projections of point clouds.

In this first approach, we were interested in the semantic segmentation of point clouds using only geometrical features. We selected a set of 5 in this study: z component, depth (distance to sensor)

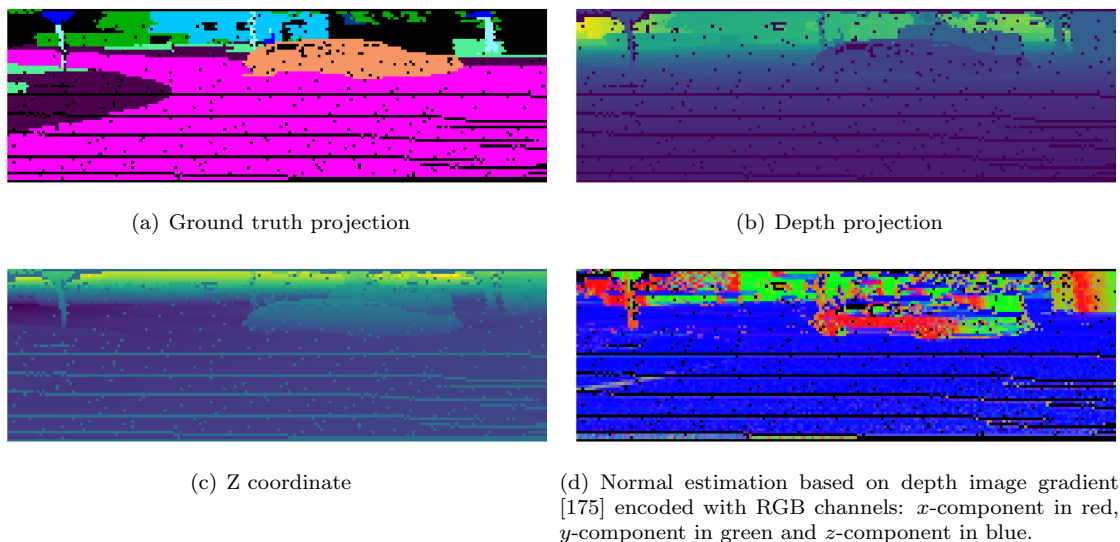


Figure 6.6: Crops of 64x200 extracted from 64x1024 spherical projections from semantic KITTI dataset.

and the 3D coordinates of the normal vector (N_x, N_y, N_z) . A straightforward normal estimation based on depth image gradient [175] is used. As a manner of example, Figure 6.6 displays features calculated from a spherical projection of an urban point cloud.

Features were selected because they have a physical meaning and were discriminant enough for some of the most frequent objects in urban datasets:

- Z coordinate: Provides information to discard objects that can be found at ground level from others, such as vegetation and buildings, against ground-like and low objects.
- Normals: Very useful to identify ground-like objects and building facades. For objects without 2D surfaces and laser beam passes through, normals do not provide helpful information. This is the case of tree leaves and some moving objects such as bikes, motorbikes, and pedestrians. This behavior is similar to dimensionality features proposed by [176].
- Depth: Inspired by results presented in RangeNet [57], we identified that they provide information about spatial proximity of objects.

Backproject to 3D data The whole learning stage is performed using 2D projections. It means that an additional step to back-project predicted labels to 3D point clouds must be included. A common strategy to assign a label to each point is to save point indexes when spherical projections are computed and then backproject predicted labels. It implies that all points projected into the same pixel will have the same label. This method is the most commonly used in BEV projections.

Recently, [57] proposed to use a KNN classifier to assign labels for unprojected points. We compared this approach varying the number of neighbors (K) against the regular approach of using indexes to give the same label to all points projected into the same pixel, using the first 300 point

IoU by class	Classes	Observations
(0, 0.1]	bicycle, bicyclist, other-ground, other-vehicle, parking, person, traffic-sign, motorcycle, motorcyclist, truck	Classes with also lowest content ratio in dataset
0.4, 0.43	pole, fence	
0.53	trunk	
0.62, 0.66	terrain, sidewalk	
0.71, 0.74, 0.74	car, building, vegetation	Vegetation is the most frequent class
0.87	road	Second most populated class in dataset

Table 6.1: Intersection over Union by class in test set of Semantic KITTI dataset.

clouds of the validation set from the Semantic KITTI dataset. Accuracy was computed for each back-projection method. Obtained results are presented in Figure 6.7.

From obtained results, one may observe that the selection of the back-projection method has a key role. In the case of $w = 1024$, labeling only the closest point to the scanner decreases about 50% accuracy compared to other methods. It occurs because scanner resolution is higher than image width, so many points are projected into the same pixel. In the case of $w = 2048$, performance is less damaged. It occurs because most of the 3D points are projected in different pixels. The most important finding of this set of experiments is that using KNN to label unprojected points has better results than assigning the same label to all the points projected into the same pixel. Comparing the accuracy of KNN varying k value, no relevant differences were observed with $k \geq 3$. This behavior was also observed by other authors such as [57].

Experiments with Semantic KITTI We performed some experiments using spherical projections with the Semantic KITTI dataset. It is a recent dataset commonly used autonomous driving applications. It is composed of urban scenes, acquired with a LiDAR scanner attached to a moving car. In Section 2.4, a detailed description of the dataset is presented.

The main goal was to identify common issues that occur in the semantic segmentation of point clouds. We tested different configurations of training parameters such as loss functions, selected features, and optimizer. In Table 6.1, we report obtained results in the test set for the best model using mean Jaccard as loss function, Adam optimizer and a set of features composed of z -component, depth, and normals. We used as image resolution $w = 1024$, $h = 64$.

Obtained results with first models using spherical projections provided interesting findings over selected configurations: **1)** Some classes are easier to learn than others such as road, building, and vegetation; **2)** Most frequent classes have the highest performance; **3)** Less populated classes have systematically the lowest performance; **4)** Classes where normals provide information have good performance even if they are distorted by projection, such as vehicles.

To deal with the minority classes, different alternatives were considered during our research: data-based approaches and loss-based approaches. In the data-based group, a simple and direct strategy is to classify using different methods of most populated and less populated classes. In the following section, we introduce a simple approach to deal with the class imbalance in point clouds datasets. The proposed methodology was developed to participate in a challenge of urban point

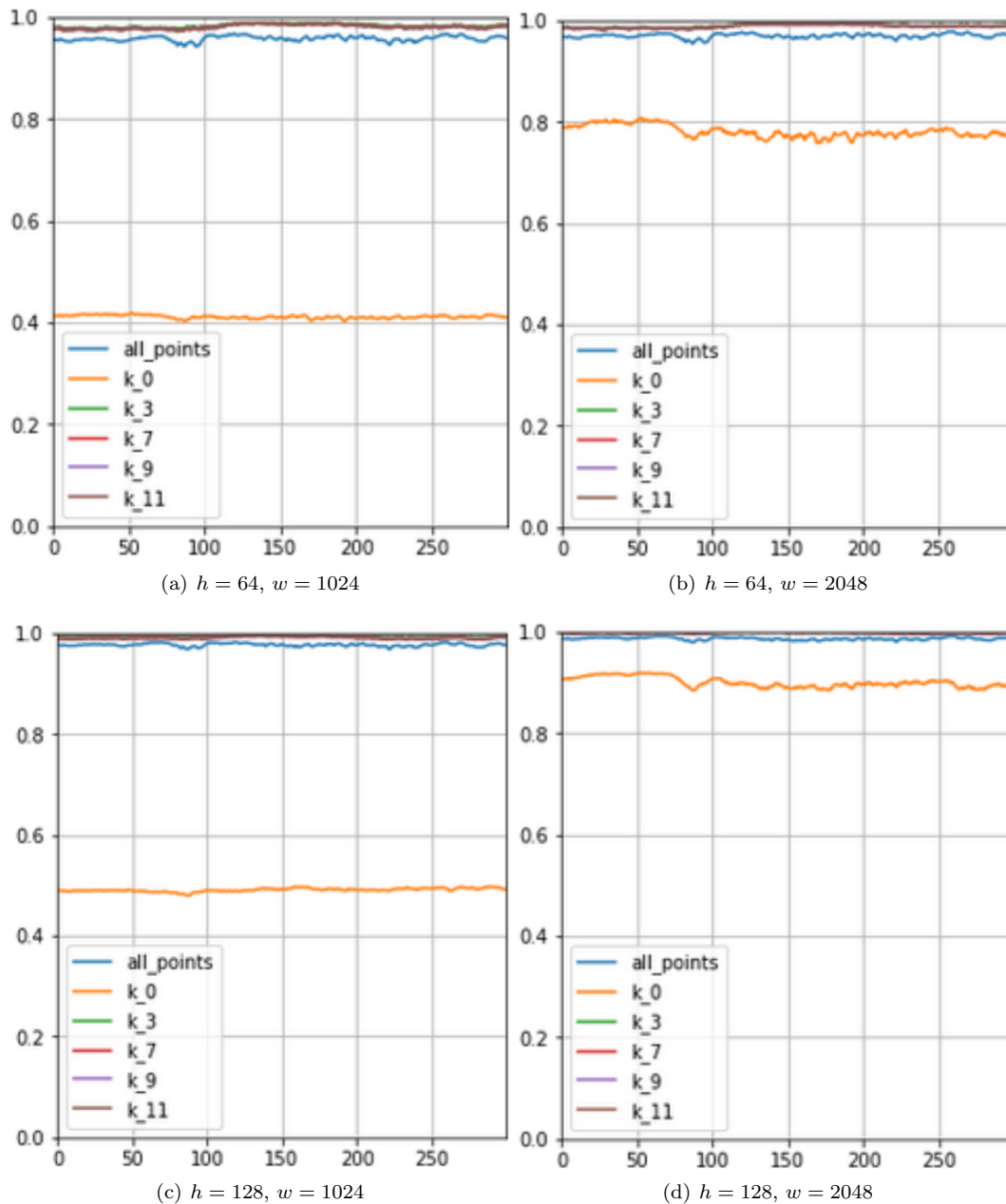


Figure 6.7: Accuracy varying back-projection method of spherical projections to 3D. Experiments were carried with first 300 point clouds from validation set of Semantic KITTI as follows: *all points*: Same label for all points projected in the same pixel; *k₀*: Label only point closest point (other points remain unlabeled); *k_n*: Perform KNN with $K = n$ to label non projected points.

Split	Undefined	Building	Car	Ground	Pole	Vegetation
Train	8.37 %	17.05 %	2.82 %	54.65 %	0.47 %	16.64 %
Validation	6.08 %	22.07 %	2.19 %	53.96 %	0.78 %	14.92 %

Table 6.2: Class rates population in Street 3D [10] dataset.

clouds segmentation, detailed in the following section.

SHREC’20 challenge We participated in a challenge to perform automatic segmentation of urban point clouds. The competition was called “SHREC 2020: 3D point cloud semantic segmentation for street scenes” and it was organized by the Department of Information and Computing Sciences from Utrecht University and Cyclomedia Technology, from the Netherlands. As part of the challenge, a paper was written by the participants and published on Computer & Graphics journal [10].

The main goal of the challenge was to perform automatic segmentation of a new dataset called Street 3D into five meaningful classes: building, car, ground, pole, and vegetation. A more detailed description of the dataset can be found in Section 2.4.2 of this document.

We proposed an automatic algorithm for point cloud segmentation using only geometrical data $[x, y, z]$. The proposed methodology is based on 2D projections. The main process relies on a spherical projection from the sensor point of view, while BEV projection is used for ground detection. We called our method *Spherical DZ Net* because it uses spherical projections of depth and z features.

The proposed algorithm is presented in Figure. 6.8 and it is mainly divided in four steps: *1)* Coarse ground detection from BEV by means of quasi-flat zones; *2)* Spherical projections and feature extraction; *3)* Train Deep Learning model with 2D images; *4)* Postprocessing and back-projection 2D predictions to 3D using a KNN classifier (with $k=3$). We name our method *Spherical DZNet* (Depth, Z and Normals network).

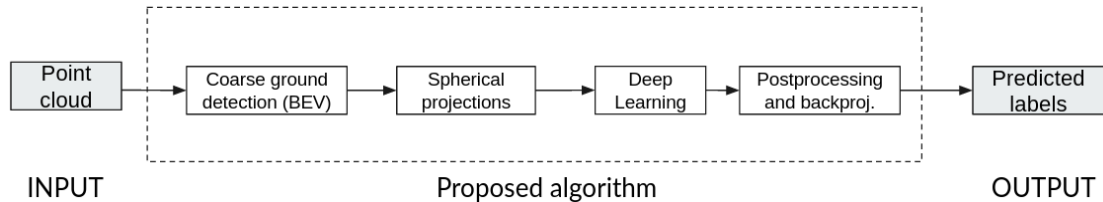


Figure 6.8: Workflow of proposed Spherical DZNet method during SHREC 2020 challenge

1) Coarse ground detection As it was identified with the first experiments using the Semantic KITTI dataset, the class imbalance is a common issue that strongly affects the model’s performance. The rate of classes in training and validation set in the Street 3D dataset is presented in Table 6.2. It is seen that *ground* class rate is considerably higher than other present classes.

A simple quasi-flat zones algorithm from the BEV projection [160] is used for this purpose, assuming that ground elevation varies smoothly. Using this approach, *ground* is classified in all point clouds from the dataset before computing spherical projections. It considerably reduces computational cost at the next steps of semantic segmentation because most of the points will be

removed, but it also implies that the ground detector must be very accurate. In this challenge, the ground detector obtained above 98 % of accuracy. We highlight that this method can be directly applied because *ground* class groups other ones such as road, sidewalk, road-line, and terrain.

2) Spherical projections Similarly, as presented before for Semantic KITTI dataset, we computed spherical projections using 3D data without points detected as ground in the previous step. It was necessary to include an additional step to estimate scanner position because the authors of the challenge did not provide it. We estimate it from the BEV as the barycentre of the empty space surrounded by the highest point density. Then, spherical projections without ground points were calculated. Figure 6.9 shows ground truth and computed features of a point cloud from the train set.

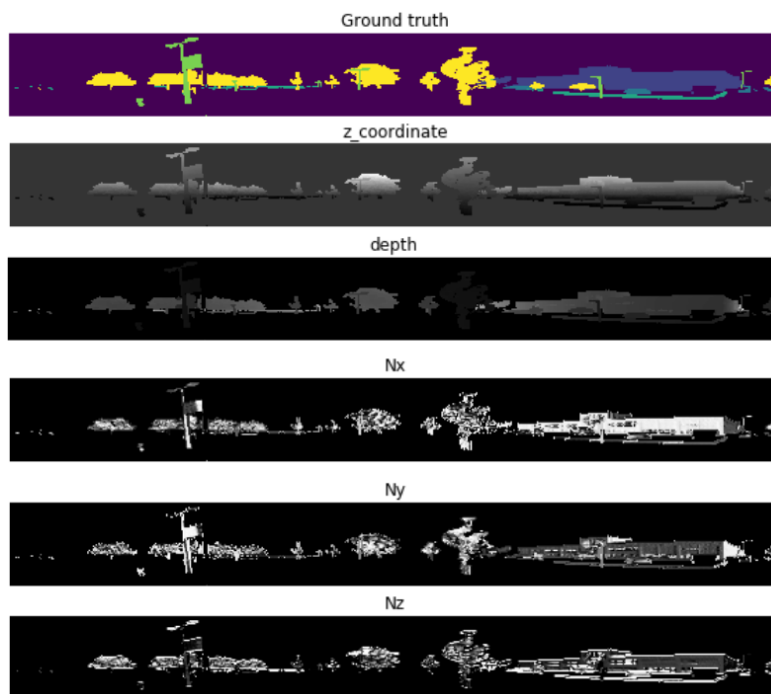


Figure 6.9: Crops of spherical projections of point cloud from Street 3D dataset.

We trained a Deep Learning model to predict the four remaining classes using the U-net based architecture [138] presented in Figure 6.10. We selected the same set of features presented before for Semantic KITTI experiments: z coordinate, depth, and normals. However, to predict semantic labels of point clouds from the test set, we compared two methods: 1) Predict labels using trained model (regular mode); 2) Remove the last layer of the model and predict labels using a KNN in a high dimensional space. This second step was performed using FAISS library [177], commonly used for efficient similarity search and clustering of dense vectors.

3) Postprocessing step To improve the classification of some points, we propose an automatic postprocessing step to correct some of the predicted labels using mathematical morphology. We performed the postprocessing using 2D projections in three steps:

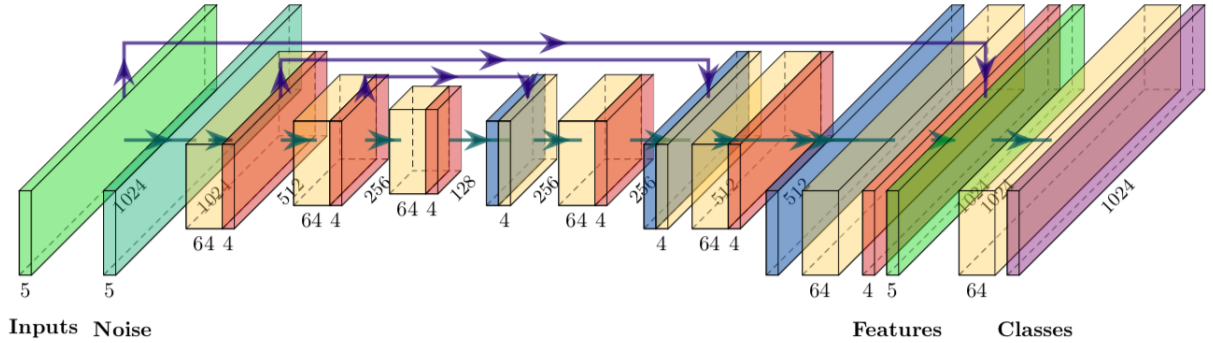


Figure 6.10: Proposed Network architecture. Yellow boxes represent Conv2D layers of 5×5 with strides (1,2), red boxes Conv2D of 1×1 followed by a dropout of 0.1. Blue color means Upsampling2D layer. Violet arrows are used to illustrate the concatenation operator. **Note:** Original input is directly concatenated to the features computed by the network.

- **Poles:** Poles are thin structures in front of other larger structures. Thus, they appear as regional minima in the depth projection. Those minima are added to the detected poles. Then, only structures with a geodesic elongation [178, 179] higher than their thickness are preserved.
- **Building:** In the BEV, structures higher than 2,5 meters and with an elongation higher than 20 are considered as buildings.
- **Small regions:** Regions classified as vegetation or building, with less than 100 pixels in the spherical view, are given the class label of the large surrounding neighbors. This is a common issue in deep learning predictions, as we present later for 3D architectures for Semantic Segmentation benchmark for PC3D.

Results In this section, we present obtained results during the challenge. As we mentioned before, different alternatives were proposed to predict labels in test set: 1) Predict using DL architecture; 2) Use KNN to predict labels instead of last layer of DL architecture; 3) Include a refinement stage to correct known errors. We report obtained results in Table 6.3.2. We highlight that the postprocessing step was performed from the labels obtained in the KNN classifier. In the fourth column, the baseline provided by the authors of the challenge.

From obtained results, we highlight: *i*) Quasi flat zones approach allows to identify *ground* with a high confidence; *ii*) Due to the working principle of LiDAR scanners, densities of the objects are strongly affected by distance. It implies that some objects like *cars* or high *vegetation* are hard to identify even by visual inspection; *iii*) Objects labeled as *pole* are not homogeneous and may group more than one single class. It also contains *traffic-sign* and *traffic-light*, for example. This may lead to overfit the model.

Also, we identified that the use of a refinement step considerably improves the performance in predicted labels from a DL architecture. This is an interesting behavior because some types of errors obtained from predictions can be easily corrected.

Class	DL	KNN	Postprocessing	Baseline
Building	0.680	0.704	0.796	0.825
Car	0.480	0.522	0.543	0.401
Ground	0.964	0.965	0.966	0.891
Pole	0.266	0.279	0.311	0.394
Vegetation	0.703	0.725	0.798	0.807
mean Iou	0.619	0.639	0.683	0.663

Table 6.3: Intersection over Union by class in test set.

Limitations of 2D based approaches From obtained results using 2D projections, some issues were identified:

- **Shapes distortion:** In spherical projections, object shapes vary according to sensor distance. It mainly affects moving objects of urban scenes such as vehicles and pedestrians. Also, due to the intrinsic definition of this type of projection, objects can be cut on image borders. This last effect affects the learning stage.
- **Occlusions:** In BEV projections, objects can hide other ones. The use of projections at different heights is a partial solution [166], but it may lead to non-desired object splitting.
- **Loss of geometrical information:** Some objects, especially smaller ones with more fine details, may lose geometrical descriptors by this issue. A straightway solution is to increase image dimensions but it creates holes in the objects and augment computational cost.
- **Our proposal obtained better performance than the baseline published by challenge authors.** Best results were obtained by a team that worked directly on 3D data using RandLA-net [180] with KPConv subsampling strategy.

Baseline for PC3D dataset

In this section, some approaches to perform semantic segmentation directly from 3D data are presented. Experiments were carried out in the proposed Paris Carla 3D dataset (PC3D), introduced in Section 2.6. We evaluated several configurations to provide the baseline benchmark for the dataset by using two known models for semantic segmentation tasks: PointNet++ [18] and KPConv [19] architectures. The final goal of this benchmark is to open a challenge for semantic segmentation of the PC3D dataset.

In this section, we present experiments performed under different configurations to demonstrate the relevance and high complexity of PC3D. We provide two baselines for all experiments with PointNet++ [18] and KPConv [19] architectures. All models were trained including color information. We report results for each point cloud of the test set and overall scores. Experiments were implemented with Torch Point 3D library [181].

One of the challenges of dense outdoor point clouds is that they can not be kept in memory due to too many points. In both baselines, we used a subsampling strategy based on spheres selection. We experimentally found that selecting spheres with a radius of $r = 10$ meters was a good compromise between computational cost and performance.

Baseline architectures We compared two DL architectures to perform semantic segmentation directly on 3D data. The first baseline is based on Pointnet++ architecture. We selected the configuration provided by the authors [18] to perform semantic segmentation in outdoor and dense environments. The second baseline is based on KPConv architecture. We selected the configuration provided by the authors for outdoor scenes [19].

Common parameters We fixed some training parameters between both baselines to compare their performance under similar conditions. Sphere centers were computed before the training stage to reduce computational cost. Using this strategy, we built a dictionary of spheres by class to feed the model with spheres of all classes. It allowed the models to see spheres of all classes at each epoch and reduce the influence of class imbalance. The number of spheres by epoch was fixed to 100, and at the end of each epoch, we selected a different point cloud from the training set. It allowed the models to learn class variability present over the dataset.

Two features were included: RGB color information and height. To prevent overfitting due to geometric information, we included data augmentation techniques: elastic distortion, random Gaussian noise with $\sigma = 0.1$, random rotations around Z, random scale anisotropic between 0.8 and 1.2, random symmetry around x and y axes. We included the following transformations to prevent overfitting due to color information: Chromatic jitter with $\sigma = 0.05$ and random dropout of RGB features with a probability of 20 %.

During training, we selected the following parameters:

- Loss function: Power Jaccard with $p = 2$ [32] for Paris data and CE + Power Jaccard with $p = 2$ [157] for CARLA data.
- Patience: 50 epochs without improvement in validation set.
- Adam optimizer with default learning rate $lr = 0.001$.

Parameters presented in this section were chosen from a set of experiments varying loss function (cross-entropy, Focal Loss, Jaccard, and Power Jaccard) and input features (RGB with z coordinate and only z coordinate). Best results were obtained with reported parameters.

Training scenarios We report obtained results in Table 6.4 with KPConv and in Table 6.5 with PointNet++. Selected data splits and a more detailed description about PC3D dataset is presented in Section 2.6. Prediction of test point clouds was performed by using a spheres-based approach using a regular grid. In this case, spheres centers were calculated with an intersection of 1/3 of its radius. Spheres radius, as done in the training set, was $r = 10$ meters.

CARLA test set We split data of CARLA to predict point clouds from the test set as follows: **1)** To predict Town 01 (urban town), we selected as training set: Town 03 and Town 05, and Town 02 as the validation set. **2)** To predict Town 07 (rural town), we selected as training set Town 04 and Town 06 as the validation set. In both baselines, we adopt the same data splits.

Paris test set We selected training and validation set according to data splits presented in Section 2.6. In both baselines, we selected the same data splits.

Class	Soufflot 0			Soufflot 3			Town 1			Town 7			Overall		
	P	R	IoU	P	R	IoU	P	R	IoU	P	R	IoU	P	R	IoU
building	78.7	96.7	76.5	98.0	99.6	98.8	64.3	78.8	54.8	54.4	75.8	46.4	59.4	77.3	69.1
fence	60.4	4.0	3.9	95.0	48.2	63.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	17.0
pedestrian	22.0	81.4	20.9	78.9	80.3	79.6	70.6	0.5	0.5	26.6	1.5	1.4	48.6	1.0	25.6
pole	48.2	87.3	45.0	76.6	80.7	78.6	18.8	93.8	18.6	14.1	77.3	13.6	16.5	85.6	38.9
road-line	82.0	44.7	40.6	87.7	59.3	70.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	27.8
road	93.1	88.7	83.1	97.5	90.8	94.0	32.3	99.8	32.3	82.6	92.7	77.5	57.5	96.2	71.7
sidewalk	60.4	96.2	59.0	59.9	98.3	74.5	66.4	4.0	3.9	13.1	11.3	6.4	39.8	7.6	36.0
vegetation	99.2	78.0	77.5	41.7	0.5	1.0	97.9	73.5	72.4	98.8	86.5	85.5	98.3	80.0	59.1
vehicles	92.7	91.2	85.0	99.8	92.4	96.0	87.2	96.1	84.2	97.2	91.3	88.9	92.2	93.7	88.5
wall	-	-	-	-	-	-	NA	NA	NA	NA	0.0	0.0	0.0	0.0	0.0
traffic-sign	55.5	74.8	46.7	75.4	89.6	81.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	32.1
bridge	-	-	-	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
rail-track	-	-	-	-	-	-	NA	NA	NA	0.0	0.0	0.0	0.0	0.0	0.0
guard-rail	-	-	-	-	-	-	0.0	0.0	0.0	42.0	74.3	36.7	21.0	37.2	18.3
traffic-light	100.0	9.5	9.5	0.0	0.0	0.0	0.0	0.0	0.0	NA	NA	NA	0.0	0.0	3.2
static	50.9	65.4	40.1	9.9	48.0	16.4	11.0	1.1	1.0	0.0	0.0	0.0	5.5	0.5	14.4
water	-	-	-	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
terrain	0.0	0.0	0.0	NA	NA	NA	0.0	0.0	0.0	93.1	78.1	73.8	46.6	39.0	24.6
mean	64.9	62.9	45.2	68.4	65.6	62.9	28.0	28.0	16.7	30.7	34.6	25.3	48.0	47.8	37.5
<i>OA</i>	84.5			94.1			66.9			89.9			83.8		

Table 6.4: Results on test set in semantic segmentation task using KPConv architecture. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.

Class	Soufflot 0			Soufflot 3			Town 1			Town 7			Overall		
	P	R	IoU	P	R	IoU	P	R	IoU	P	R	IoU	P	R	IoU
building	39.1	96.0	38.4	94.0	98.7	92.8	20.4	65.1	18.4	0.0	0.0	0.0	38.4	64.9	37.4
fence	0.0	0.0	0.0	40.3	35.9	23.4	0.0	0.0	0.0	0.0	0.0	0.0	10.1	9.0	5.9
pedestrian	12.3	29.3	9.4	54.5	23.4	19.6	0.0	0.0	0.0	0.0	0.0	0.0	16.7	13.2	7.3
pole	55.2	25.9	21.3	68.5	29.0	25.5	0.0	0.0	0.0	0.0	0.0	0.0	30.9	13.7	11.7
road-line	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
road	95.2	76.6	73.7	96.4	84.5	81.8	0.0	0.0	0.0	32.3	89.8	31.1	56.0	62.7	46.7
sidewalk	34.5	90.3	33.2	32.4	91.0	31.3	0.0	0.0	0.0	0.0	0.0	0.0	16.7	45.3	16.1
vegetation	99.7	0.0	0.0	0.0	0.0	0.0	87.8	77.1	69.6	61.7	83.9	55.2	62.3	40.3	31.2
vehicles	57.1	4.8	4.6	87.0	35.8	33.9	0.0	0.0	0.0	0.0	0.0	0.0	36.0	10.1	9.6
wall	-	-	-	-	-	-	0.0	0.0	0.0	-	-	-	0.0	0.0	0.0
traffic-sign	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bridge	-	-	-	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
guard-rail	-	-	-	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
traffic-light	0.0	0.0	0.0	0.0	0.0	0.0	-	-	-	-	-	-	0.0	0.0	0.0
static	23.8	0.3	0.2	1.6	7.7	1.3	0.0	0.0	0.0	0.0	0.0	0.0	6.4	2.0	0.4
water	-	-	-	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
terrain	57.1	0.0	0.0	-	-	-	0.0	0.0	0.0	0.0	0.0	0.0	19.0	0.0	0.0
mean	36.5	24.8	13.9	39.6	33.8	25.8	6.8	8.9	5.5	6.3	11.6	5.8	22.3	19.8	12.7
<i>OA</i>	71.5			84.4			33.6			35.2			56.2		

Table 6.5: Results on test set in semantic segmentation task using PointNet++ architecture. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.

6.3.3 Transfer learning

TL was performed to demonstrate the importance of synthetic point clouds generated in CARLA to perform semantic segmentation in real-world point clouds. We selected the model with the best performance in point clouds of the test set from CARLA data. Then, we took it as input in the training stage with Paris data.

We carried out different types of experiments as follows: 1) Predict test point clouds of Paris data using best model obtained in urban towns from CARLA without training in Paris data (*no training*); 2) Freeze whole model except last layer; 3) Freeze feature extractor of the network; 4) No frozen parameters; 5) Training a model since scratch using only Paris data. For experiments 2, 3, and 4, we took a pre-trained model with CARLA data. These scenarios were selected to evaluate the relevance of learned features in CARLA and their capacity to discriminate classes in Paris data. We report a summary of different experiments in Table 6.6. Best results using TL were obtained in scenario 4: model pre-trained in CARLA without frozen parameters during training with Paris data and they are presented in Table 6.7.

Transfer learning scenario	Paris		Overall
	S_0	S_3	
No fine-tuning	20.6	17.7	19.2
Freeze except last layer	24.1	31.0	27.6
Freeze feature extractor	29.0	41.3	35.2
No frozen parameters	42.8	50.0	46.4
No transfer	45.2	62.9	51.7

Table 6.6: Results in transfer learning for semantic segmentation task using KPConv architecture on our dataset Paris-CARLA-3D. Results are mIoU in %. Pre-training was done using urban towns from CARLA. *no-training*: model pretrained with CARLA data, without training in Paris data; *no-frozen parameters*: model pretrained with CARLA without frozen parameters during training with Paris; *no-transfer*: model trained since scratch with Paris data.

From obtained results in studied scenarios, the relevance of synthetic data in this type of task is observed. From Table 6.6, one may observe that performance of *no frozen parameters* and *no transfer* are in the same order of magnitude (46.4% vs 51.7 %). It is a relevant finding it allows us the use of synthetic data in a pre-training stage and to reduce training time with real-world data. For example, we evidenced that the duration of training stage in *no-frozen parameters* was 25 % shorter than in *no-transfer*. A second finding was that a model trained with synthetic data can not be directly applied to real-world data (*no-training* row). It is an expected result because objects and class distributions in CARLA towns are different from real-world ones. Several alternatives are now opened to close the existing gap between synthetic and real data, such as Domain Adaptation methods.

6.4 Hierarchical learning for semantic segmentation

In this section, we present proposed strategies to include hierarchies in urban point clouds segmentation. Firstly, by using a two-steps learning approach to training different models, each one is specialized into a subset of classes from all available ones. Secondly, utilizing a tree representation of classes embedded in the learning process of DL architecture. Obtained results demonstrate

Class	Soufflot 0			Soufflot 3			Overall		
	P	R	IoU	P	R	IoU	P	R	IoU
building	75.79	98.54	74.90	97.16	99.69	96.80	86.48	99.12	85.85
fence	84.95	11.52	11.20	87.40	31.23	29.80	86.18	21.38	20.50
pedestrian	25.60	75.51	23.60	74.08	71.86	57.40	49.84	73.69	40.50
pole	63.09	83.28	55.90	88.51	74.84	68.20	75.80	79.06	62.05
road-line	84.18	51.86	47.20	87.55	71.76	65.10	85.87	61.81	56.15
road	91.80	95.81	88.20	95.55	94.70	90.60	93.68	95.26	89.40
sidewalk	74.64	87.42	67.40	73.32	88.99	67.20	73.98	88.21	67.30
vegetation	99.59	85.09	84.70	72.37	4.45	4.30	85.98	44.77	44.50
vehicles	92.28	81.09	75.90	98.98	85.95	85.10	95.63	83.52	80.50
wall	0.00	0.00	0.00	-	-	-	0.00	0.00	0.00
traffic-sign	32.87	29.06	18.20	54.36	39.65	29.70	43.62	34.36	23.95
traffic-light	99.65	6.89	6.80	0.00	0.00	0.00	49.83	3.45	3.40
static	74.79	51.61	43.90	5.59	64.94	5.40	40.19	58.28	24.65
terrain	85.94	0.73	0.70	-	-	-	85.94	0.73	0.70
mean	70.37	54.17	42.76	69.57	60.67	49.97	69.97	57.42	46.36
<i>OA</i>	87.64			93.90			90.77		

Table 6.7: Results on test set of Paris data in semantic segmentation task with transfer learning from CARLA data. P: Precision, R: Recall, IoU: Intersection over Union, mIoU: Mean IoU, OA: Overall Accuracy.

that using a different approach for a highly imbalanced dataset improves performance in semantic segmentation tasks.

6.4.1 Related work

The use of hierarchical approaches during learning is a field that has attracted the interest of researchers over the last years. As most of the methods developed for automatic tasks in point clouds, hierarchical studies were initially proposed for 2D images. Many works and literature have been published in the last 40 years for semantic segmentation tasks using hierarchical concepts. One of the most used and successful techniques is the use of graphs to build a hierarchical representation of images. It allows to group similar pixels into regions according to some similarity criteria such as color and texture. A detailed survey about most relevant methods until 2004 is stated by [182]. In the same spirit of representing images using graphs, Graph Neural Networks (GNN) have considerably attracted research in many different image processing tasks in the last few years. A survey with relevant methods using GNN in Deep Learning applications is introduced by [183].

A second alternative to use hierarchies in automatic tasks is to represent classes structure directly and not only for input data. By using this perspective, the learning process itself can be modified because some *a priori* information is included in the model. For example, in [184] classes of histopathology classes are represented using a hierarchical tree. In this case, they proposed training different classifiers and learning “*n* classes vs the rest classes” at two hierarchical levels. This method improves performance compared to a plain representation of classes, especially in high imbalance datasets. A similar approach is presented by [185], where they propose to use a set of

binary classifiers to perform image classification based on a hierarchical tree. Each classifier learns to distinguish between a pair of objects from the tree hierarchy.

A third alternative, related to the previous one, is to represent class hierarchies and embed them into the cost function. With this technique, the learning process can penalize the classes differently according to their hierarchical representation. For example, in [186] is presented a loss function that establishes ultrametric distances for every class. In this approach, the model penalizes harder more *dangerous* errors in the context of breast cancer detection in histopathological images. More recently, a hierarchical representation of classes for semantic segmentation of human faces was proposed by [187]. They grouped classes using a tree of four depth levels, grouping classes according to their physical meaning. Obtained results with hierarchical loss were better than using a regular loss.

Other alternatives to represent hierarchies in image processing are those related to attention mechanisms. This concept has attracted the deep learning research community because it seeks to address the focus of the model according to the type of application. A recent survey and classification of attention mechanisms in deep learning is available at [188].

6.4.2 Learning based on a class hierarchy

We studied two alternatives to represent hierarchies in semantic segmentation of point clouds: data-based and loss-based methods. The first group is a direct extension of the proposed algorithm for the SHREC'20 challenge, described in Section 6.3.2, but directly with 3D data. The second group extends the proposed power Jaccard loss for image segmentation with a hierarchical approach. We introduce proposed approaches as well as obtained results using Paris data from the PC3D dataset below. We focused on Paris data to work with real-world point clouds.

Specialized models

In the same spirit of the method proposed for the SHREC'20 challenge, we studied the advantages of using specialized models to learn a subset of classes according to the type of application. This approach is inspired by the *second alternative* methods introduced in Section 6.4.1. We seek to simplify the learning stage and improve the performance of semantic segmentation tasks by training two or more models specialized for different subsets of classes. For example, in point clouds from urban scenes, *ground-like* classes such as road, sidewalk, and terrain are much more similar between them than compared to *vertical-like* objects such as poles. However, class similarities described above cannot be directly embedded using a single model to learn all classes.

From Table 2.2, it is observed that ground-like classes and building class represent almost 80 % of the points from the dataset. These classes also have the highest performance in tested scenarios, as presented in Section 6.3. This data imbalance has a significant impact on the models, and it prevents to learn classes that should be easy to distinguish. For example, fence vs road or sidewalk vs vehicle.

In this section, we compared three scenarios to evaluate the influence of reducing the number of classes by removing the most frequent ones:

1. Learn all classes together using DL model (M1)
2. Discard ground-like classes including road, sidewalk, road-line and terrain. Learn the rest of the classes (8) using a DL model (M2).

3. Discard ground-like classes and building classes. Learn the rest of the classes (7) using a DL model (M3).

In all scenarios, we selected KPConv model with the following training parameters: Z + RGB as features, Adam optimizer, loss function as the sum of cross-entropy and power Jaccard with $p = 2$. Obtained results are presented in Table 6.8.

Class	Soufflot 0			Soufflot 3		
	M1	M2	M3	M1	M2	M3
building	76.50%	58.15%	-	97.50%	97.90%	-
fence	3.90%	13.55%	31.82%	46.90%	46.53%	90.22%
pedestrian	20.90%	44.31%	39.21%	66.10%	51.18%	74.17%
pole	45.00%	58.79%	38.02%	64.70%	68.78%	71.32%
road-line	40.60%	-	-	54.70%	-	-
road	83.10%	-	-	88.60%	-	-
sidewalk	59.00%	-	-	59.30%	-	-
vegetation	77.50%	53.47%	93.48%	0.50%	0.00%	81.66%
vehicles	85.00%	84.73%	84.04%	92.20%	87.35%	92.63%
traffic-sign	46.70%	30.06%	40.86%	69.20%	55.27%	51.94%
traffic-light	9.50%	26.77%	25.75%	0.00%	1.15%	3.64%
static	40.10%	66.58%	45.71%	8.90%	10.76%	7.53%
terrain	0.00%	-	-	-	-	-
<i>mean</i>	45.22%	48.49%	49.86%	54.05%	46.55%	59.14%

Table 6.8: IoU by class of Paris data in semantic segmentation task without most frequent classes. We report only IoU by class. M1: Learning all classes; M2: Without ground-like classes; M3: Without ground-like and building classes. *mIoU* is the mean IoU of learned classes.

From obtained results with these two experiments, it was observed that removing the most frequent classes allows the model to focus on less populated and hard classes in the dataset. It also demonstrates that class imbalance strongly affects optimization during learning. This finding confirms the interest in using a different approach for learning subsets of classes by grouping classes into a simplified representation based on their geometric similarity, for example. An alternative is to use the “ n classes vs the rest classes” and then assembly the predictions of the different models. Another one is to embed some *a priori* information about the classes in the learning stage. We opted for the second option and present it in the following section.

Hierarchical tree

Similarly, as our proposed power Jaccard loss in Chapter 5 for semantic segmentation of images, we studied a strategy to penalize the errors differently during learning. A way to achieve it, is through a class representation that provides more information about related and unrelated classes. For example, using the loss function, it is possible to penalize stronger if the model miss-predicts a pole with a building than if it confuses it with a traffic light.

We propose to build a hierarchical tree to group similar classes from urban street scenes in point clouds, as it is presented in Figure 6.11. This representation allows to include *a priori* about the classes and to penalize harder certain types of errors during learning.

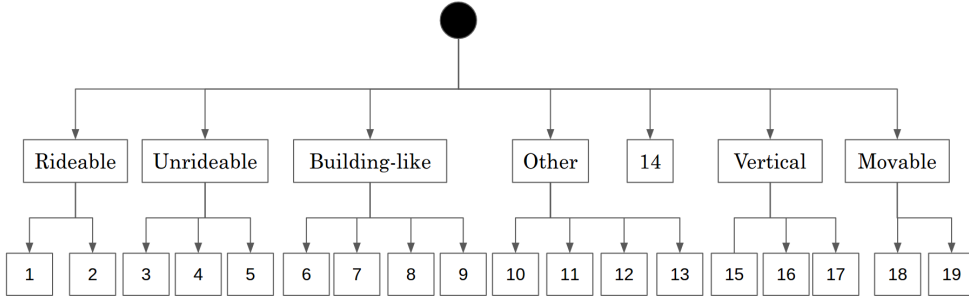


Figure 6.11: Hierarchical representation of urban point clouds from Paris data (expert knowledge). First level is composed of 7 nodes: rideable, unrideable, building-like, other, vegetation (14), vertical and movable. Second level represents the leaves of the tree: 1 road, 2 road-line; 3 ground, 4 terrain, 5 sidewalk; 6 building, 7 wall, 8 fence, 9 bridge; 10 rail-track, 11 guard-rail, 12 water, 13 static; 15 pole, 16 traffic-sign, 17 traffic-light; 18 pedestrian, 19 vehicle.

The proposed tree has two hierarchy levels: in the first level, classes are grouped according to their geometrical similarity; in the second level, they are grouped according to the type of use in the context of the autonomous vehicle. Not every class has two levels of hierarchy, as vegetation, for example. Using this hierarchical representation of classes, we propose to embed it into a loss function to penalize errors differently during training.

The proposed loss is the sum of the loss functions at each hierarchical level as follows:

$$L_{total} = L_{nodes} + L_{leafs} \quad (6.2)$$

In the L_{nodes} level, the loss is calculated at the first level of hierarchy. It is performed in two steps: first, activation values of the siblings of each node must be computed; then, a classical procedure to compute a loss function can be performed. The calculation of L_{leafs} on the second level is straightforward. It is necessary only to take the activation values of each class and to perform the classical procedure for loss function computation.

In Algorithm 3 is presented the pseudo-code of our proposal. We implemented two of the most used deep learning frameworks in Python: Tensorflow and Pytorch. The proposed hierarchical loss function has the following inputs: 1) Y_{true} : Ground truth vector.; 2) Y_{pred} : Predicted vector.; 3) $Tree$: Dictionary with mapping of classes to nodes.; 4) $Base\ loss$: Selected loss to compute at each level. The output of the algorithm is a scalar value of L_{total} .

Using the proposed hierarchical loss, we used power Jaccard with $p = 2$ as *base loss* and compared their performance with other loss functions. We report obtained results in Table 6.9 for Soufflot 0 and in Table 6.10 for Soufflot 3.

From results presented in Tables 6.9 and 6.10, one may observe that the selection of the loss function has a very important role in models performance. For example, CE (Crossentropy) has the lowest mean IoU in both point clouds. However, when CE is summed with power Jaccard, mean IoU increases 5 % in Soufflot 0 and 7 % in Soufflot 3. This demonstrates why in some competitions, participants very often use a combined loss as a cost function.

The models trained with the proposed hierarchical representation obtained the highest performances in both cases. In the case of Soufflot 0, the most significant improvement occurred in vegetation. In this class, the other models confused top vegetation with building class very often.

Input : Y_{true} , Y_{pred} , Tree, base loss
for each node in Tree do
 | y_{true} = Sum siblings ground truth of Y_{true} ;
 | y_{pred} = Sum siblings activation values of Y_{pred} ;
end
 L_{nodes} = Compute *base loss* with y_{true} , y_{pred} in nodes ;
 L_{leaves} = Compute *base loss* with Y_{true} , Y_{pred} in leaves ;
 $L_{total} = L_{nodes} + L_{leaves}$
Output: L_{total}

Algorithm 3: Proposed hierarchical loss function

Class	Rate	CE	CE + Jac 2	Jac 2	Hierar.
building	15.38%	54.19%	76.57%	79.39%	82.15%
fence	2.36%	9.40%	3.92%	2.55%	1.46%
pedestrian	0.23%	32.28%	20.90%	49.21%	33.12%
pole	0.58%	43.34%	45.03%	53.29%	49.12%
road-line	3.91%	44.39%	40.68%	39.87%	44.39%
road	42.29%	78.41%	83.18%	83.04%	87.09%
sidewalk	10.42%	51.92%	59.01%	60.66%	69.08%
vegetation	19.08%	41.04%	77.54%	86.97%	92.19%
vehicles	1.31%	73.71%	85.04%	76.25%	74.57%
traffic-sign	0.15%	23.44%	46.76%	27.74%	45.22%
traffic-light	0.13%	23.28%	9.50%	13.98%	22.87%
static	2.72%	52.06%	40.11%	49.50%	51.10%
terrain	1.44%	0.00%	0.00%	0.40%	0.00%
<i>mIoU</i>	-	40.57%	45.25%	47.91%	50.18%
OA	-	81.32%	84.47%	86.20%	87.46%

Table 6.9: Performance using IoU in Soufflot 0 varying loss function and using hierarchical-based approach. First column displays class names; second column, rate of points by class; CE is categorical crossentropy; CE + Jac 2, states for the sum of categorical crossentropy and power Jaccard with $p = 2$; Jac. 2 is power Jaccard with $p = 2$; Hierarc. is the hierarchical loss with Jac 2 as *base loss*

A possible explanation of this type of error is that the top of the trees has low points density as the top of the building. Additionally, the z component of the points belonging to these classes is not discriminant enough to differentiate between them. Figure 6.12 displays a comparison of the predicted labels in Soufflot 0 in a region with trees.

Concerning Soufflot 3, hierarchical loss obtained the highest mean IoU and the highest IoU in 6 out of 12 classes. In some cases, such as pedestrian and sidewalk, the improvement is close to 30 % compared to CE. This is a very important finding that demonstrates that using a representation that includes *a priori* information improves the performance. In our case, this information was included with the hierarchical tree displayed in Figure 6.11.

Class	Rate	CE	CE + Jac 2	Jac 2	Hierar.
building	36.59%	97.66%	97.56%	97.62%	97.70%
fence	0.82%	46.28%	46.97%	52.28%	53.79%
pedestrian	1.01%	46.35%	66.10%	69.40%	72.23%
pole	0.79%	61.23%	64.73%	69.47%	66.75%
road-line	4.09%	57.76%	54.72%	57.85%	60.44%
road	37.59%	77.98%	88.69%	84.96%	92.18%
sidewalk	6.68%	42.07%	59.31%	52.50%	72.70%
vegetation	0.30%	0.00%	0.51%	0.48%	0.14%
vehicles	6.47%	82.55%	92.26%	90.68%	86.23%
traffic-sign	0.11%	52.69%	69.28%	57.26%	69.05%
traffic-light	0.06%	0.00%	0.00%	0.13%	0.00%
static	0.06%	3.98%	8.94%	9.05%	4.63%
terrain	-	-	-	-	-
<i>mIoU</i>	-	47.38%	54.09%	53.47%	56.32%
OA	-	92.59%	94.06%	93.70%	94.87%

Table 6.10: Performance using IoU in Soufflot 3 varying loss function and using hierarchical-based approach. First column displays class names; second column, rate of points by class; CE is categorical crossentropy; CE + Jac 2, states for the sum of categorical crossentropy and power Jaccard with $p = 2$; Jac. 2 is power Jaccard with $p = 2$; Hierarc. is the hierarchical loss with Jac 2 as *base loss*

6.5 Instance segmentation

One of the main interests of proposing Paris Carla 3D was to create a challenging dataset for different automatic tasks for autonomous driving, such as Semantic Segmentation (SS), scene completion (SC), and Instance Segmentation (IS). Previously, in Section 6.3, we introduced the benchmark for SS. Now, in this section, we present our proposal to perform IS in the created dataset.

6.5.1 Related Work

In this section, we present some related works in the context of instance segmentation of urban point clouds. Similarly, as presented in Section 6.3.1 for semantic segmentation, object detection methods can be grouped according to the way they represent point clouds. As proposed by [189], most of these methods can be grouped on: 2D projections; frustum-based; voxel-based; pillar-based; point-based. Then, different strategies to extract features and detect objects on the scene have been proposed according to the type of data representation..

The first group of 2D projections represents point clouds in structured grids as presented before for semantic segmentation applications (Section 6.3.2). For object detection tasks, the most common projections are BEV and Spherical Projection, also known as Front View projection (FV). For example, the projection of the distance to the sensor in spherical projection is also called Range View (RV) projection. Some works that rely on FV and RV are VeloFCN [190] and MultiView3D (MV3D) [191]. Due to the characteristic of this type of projection where object shapes are strongly affected by its distance to the sensor, the best results have been obtained by using BEV projections.

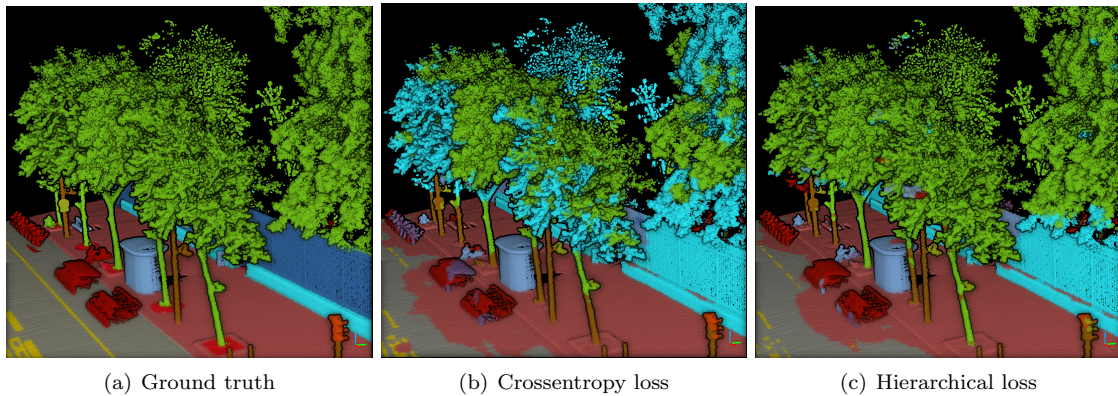


Figure 6.12: Comparison of predicted labels in Soufflot 0 varying the loss function. The model trained with hierarchical loss can discriminate top vegetation from building. Other models get systematically confused between top vegetation and building class.

Using this approach, several works using non-learning methods such as [54] and [160], and learning methods such as PIXOR [192], AVOD [193] and LaserNet++ [194], have been proposed.

The second group based on frustums divides 3D data on solids bounded by parallel planes. This approach detects objects on RGB images and then projects them to 3D data based on frustum representation. Some of the methods that use this approach are: Frustum point net [195], Frustum ConvNet [196] and SIFRNet [197].

The third group relies on a voxel representation. The main limitation of this approach is the computational cost related to selected voxel size and the loss of resolution. It may limit the detection of small objects. Some methods based on voxel representation are: PVRCNN [198], VoxelNet [63] and Voxel FPN [199].

The fourth group represents 3D data in vertical columns called pillars. It allows defining bins in a horizontal plane and then extracts features from them. The most successful method using this approach is PointPillars [200].

The fifth group based directly on 3D points works directly with raw point clouds. This approach has attracted most of the attention in the last years due to its capability to represent local and global features of point clouds and the profit of geometrical information available on 3D data. Several methods have been proposed, such as well known PointNet [17] and PointNet++ [18], IPOD [201], Fast Point RCNN [202] and PointPainting [203].

Additionally to previous works proposed for object detection and counting, recent methods seek to directly perform instance segmentation from point clouds. Some of those methods are: Panoster [204], Instance-polar Net [205], Instance deep lab [206] and JSIS3D [207]. By using FV projection, in the same spirit of RangeNet++ [57], same authors recently proposed [208].

6.5.2 Baseline for PC3D dataset

In this section, a baseline for Instance Segmentation for the PC3D dataset and its evaluation with introduced metrics is presented. A hybrid approach combining deep learning and mathematical morphology to predict instance labels is proposed. The main goal was to introduce this novel

and complex task and demonstrate its relevance for autonomous driving and mobile cartography applications.

As presented by [54], urban objects can be classified using geometrical and contextual features. In our case, we start from already predicted *cars* and *pedestrians* presented in Table 6.4. Then, instances are detected by using BEV projections and mathematical morphology.

We computed the following BEV projections and selected in all cases a pixel resolution of 10 cm:

- Occupancy image (I_b): Binary image with presence or not of *things* class.
- Elevation image (I_h): Stores the maximal elevation among all projected points on the same pixel.
- Accumulation image (I_{acc}): Stores the number of points projected on the same pixel.

Vehicles This class is composed of different objects such as cars, motorbikes, bikes, and scooters. Additionally, it also includes moving and parked vehicles that make it harder to determine object boundaries. This variability makes it challenging for automatic tasks.

From BEV projections, vehicles detection is performed as follows:

1. Discard the points predicted as vehicle if z component is greater than 4 m in I_h relative to DEM (Digital Elevation Model).
2. Connect close components with a morphological dilation of I_b by a square of 5 pixels size.
3. Fill holes smaller than 10 pixels inside each connected component. It is performed with a morphological area closing.
4. Discard instances with less than 500 points in I_{acc} .
5. Discard instances not surrounded by ground-like classes in I_b .

Pedestrians As introduced before for vehicles, the pedestrians class may contain moving objects. It implies that object boundaries are not always well defined and it may lead to some cases where several instances can be mixed.

From results presented in tables 6.4 and 6.5, it can be seen that models trained only with rural towns were not able to learn pedestrian class in Town 07. It occurs because, in this type of town, pedestrians are even less frequent than in urban towns, as it is shown in Table 2.1. To tackle this problem, we trained a model with all towns obtaining better performance in some classes, such as pedestrians and rail-track. We used predictions of this model as input for detecting pedestrian instances.

We followed a similar approach as described before for vehicle instances based on BEV projections. The steps to detect pedestrian instances are:

1. Discard pedestrian points if z component is greater than 3 m. in I_h .
2. Connect close components and fill small holes as it is described for vehicle class.
3. Discard instances with less than 100 points in I_{acc} .

4. Discard instances not surrounded by ground-like classes in I_b .

Then, instance labels of BEV images are back-projected to 3D data to provide point-wise predictions. In Table 6.11 we report obtained results in Instance Segmentation using proposed approach.

	SM	PQ	$mIoU$	# instances
Soufflot 0 - Vehicles	90%	70.92%	81.56%	10
Soufflot 3 - Vehicles	32.56%	40.48%	28.01%	86
Town 1 - Vehicles	17.07%	20.44%	14.21%	41
Town 7 - Vehicles	74.07%	72.55%	61.17%	27
Town 1 - Pedestrians	18.37%	17.01%	13.89%	49
Town 7 - Pedestrians	100%	8.99%	65.98%	3

Table 6.11: Results on test set in instance segmentation task. SM: Segment Matching, PQ: Instance Quality, $mIoU$: Mean IoU

6.6 Color in urban point clouds

The analysis presented in Section 4.3 demonstrates that the current technology of HS cameras does not allow the acquisition of good quality images to improve scene description in urban scenes. As an alternative to evaluate the influence of spectral information in this type of application, we conducted a study using our new Paris Carla 3D dataset. It was selected because it is composed of colored point clouds of the same neighborhood where HS images of the REPLICa project were acquired.

The results presented in Chapter 6 evidence that models trained with color features were better than the models trained only with geometrical features. However, the improvement did not occur for every class and every scenario. To determine in what classes the color systematically enhances the performance, we carried out a comparative analysis using models trained with and without color features and varying the number of learned classes, as presented before in Section 6.4.2.

6.6.1 Experiments

The experiments were conducted using the same training parameters described in Section 6.3.2 and KPConv architecture, which demonstrated better results than PointNet++ in all evaluated cases. However, in this set of experiments, we selected as a loss function our proposed Power Jaccard with $p = 2$ [157]. To prevent overfitting with color information, a random dropout of RGB features with a probability of 20 % was chosen in every experiment.

We performed the following experiments varying the number of learned classes to identify when RGB information plays a significant role. The scenarios where fewer classes were learned removes the class imbalance of the dataset that may impact learning and let us focus on the influence of the color for different classes:

1. 13 classes: Discarding ground-like classes (ground, terrain, sidewalk, road, road-line) and building class. We called this scenario P_1 .
2. 14 classes: Discarding ground-like classes. We called this scenario P_2 .

3. Learning all classes. We called this scenario P_3 .
4. Hierarchical learning introduced in Section 6.4. We called this scenario P_4 .

For each scenario, we trained two models using the following input features: $Z + \text{RGB}$; only Z . Then, we compared the obtained results in the test set by using IoU as the evaluation metric. We evaluated the four described scenarios using data splits proposed for Paris Carla 3D dataset: S_1 and S_2 for training, S_4 for validation and S_0 and S_3 for testing. Obtained results in test set are reported in Table 6.12.

6.6.2 Analysis of results

In this section, we present an analysis by class based on the performance presented in Table 6.12 in the test set of Paris data composed of S_0 and S_3 . However, results must be carefully analyzed because the test set contains two very different point clouds. On the one hand, S_0 was acquired close to the entrance of the Luxembourg Gardens and it is composed of different types of vegetation and ground-like classes. Moreover, it also has wide streets including cycling and bus paths. On the other hand, S_3 is a *most typical* urban environment with a straight road enclosed by tall buildings without trees. Present vegetation in S_3 is located on the flower pots of some balconies. These differences must be taken into account when comparing the performance of vegetation and terrain classes because their rate of points varies a lot between both test point clouds (see Table 2.2) and the type of present objects are very different. Additionally, ignored classes are excluded from the evaluation process in the scenarios where fewer classes were learned (P_1, P_2).

Building: Present in 3 out of 4 scenarios. For S_0 , the best performance was obtained in P_4 with RGB features. For S_3 , all the models obtained around 97 % of IoU, with or without RGB features. It is an expected result because present buildings have many different colors. An alternative to having a scenario where color may help is to have a more detailed classification of the objects including walls, windows, and doors, for example. In this case, color should be useful due to the architectural characteristics of the buildings presented in Paris data.

Fence: Present in all scenarios. For both test point clouds, the best performance was obtained in P_1 with RGB features. We must highlight that in the P_1 scenario, the building class is not learned and it permits obtaining the highest performances, with or without RGB features. It occurs because most of the points of this class are part of the grid enclosing Luxembourg gardens located above a stone structure labeled as a building. The use of RGB features improves the performance in most cases and it is an expected result because grids in Paris data are usually black.

Pedestrian: Present in all scenarios. For S_0 , the best performance was obtained in P_2 without RGB features. For S_3 , the best performance was obtained in P_1 without RGB features. In this class, RGB features do not improve performance. It is an expected result because pedestrians can have any color.

Pole: Present in all scenarios. For S_0 , the best performance was obtained in P_2 without RGB features. For S_3 , the best performance was obtained in P_1 with RGB features. In S_0 , most of the present poles are black or dark grey and have a vertical-like geometry. However, a common error is

Class	S_0				S_3										
	W.	NO	W.	NO	W.	NO	W.	NO							
building	-	-	58.15	80.48	79.39	74.24	82.15	72.09	-	97.9	97.92	97.62	97.14	97.7	97.34
fence	31.82	22.54	13.55	16.46	2.55	0.0	1.46	7.04	90.22	76.03	46.53	52.68	49.82	53.79	44.57
pedestrian	39.21	28.18	44.31	62.94	49.21	61.77	33.12	28.03	74.17	82.81	51.18	80.26	69.4	74.3	72.23
pole	38.02	37.8	58.79	44.89	53.29	46.59	49.12	37.66	71.32	73.64	68.78	67.55	69.47	61.93	66.75
road-line	-	-	-	-	39.87	0.0	44.39	0.0	-	-	-	-	57.85	0.0	60.44
road	-	-	-	-	83.04	75.72	87.09	81.75	-	-	-	-	84.96	71.35	92.18
sidewalk	-	-	-	-	60.66	46.56	69.08	56.21	-	-	-	52.5	38.88	72.7	63.23
vegetation	93.48	91.26	53.47	84.01	86.97	84.99	92.19	71.55	81.66	81.73	0.0	0.0	0.48	0.0	0.14
vehicles	84.04	74.3	84.73	66.77	76.25	64.22	74.57	69.62	92.63	92.68	87.35	90.85	90.68	73.13	86.23
traffic-sign	40.86	32.25	30.06	20.85	27.74	13.31	45.22	13.1	51.94	49.04	55.27	36.64	57.26	29.58	69.05
traffic-light	25.75	13.61	26.77	14.61	13.98	0.0	22.87	26.99	3.64	9.62	1.15	0.0	0.13	0.0	0.0
static	45.71	28.54	66.58	33.2	49.5	45.29	51.1	44.81	7.53	9.44	10.76	6.91	9.05	1.99	3.48
terrain	-	-	-	-	0.4	0.0	0.0	0.0	59.14	-	-	-	-	-	-
mean	49.86	41.06	48.49	47.13	47.91	39.44	50.18	39.14	59.14	59.37	46.55	48.09	53.47	41.51	56.32

Table 6.12: IoU in test set of Paris data with different training scenarios: P_1) 13 classes; P_2) 14 classes; P_3) All classes; P_4) Hierarchical learning. In each scenario, two models were trained with RGB features (W) and without RGB features (NO). Bold text is the highest value by class by test point cloud (S_0 and S_3).

to confuse them with tree trunks. An alternative to correct the confusion between tree-trunk and pole is to include a postprocessing step as presented in Section 2.5.1. In S_3 , the performance of different models is very similar in all cases with or without color features and it demonstrates that most of the poles can be distinguished by their geometry.

Road-line: Present in 2 of 4 scenarios. This class can only be learned when RGB features are included. In both point clouds, the best results were obtained in the P_4 scenario.

Road: Present in 2 out of 4 scenarios. For both test point clouds, the best performance was obtained in P_4 with RGB features. In this class, RGB features provide helpful information to improve performance. These results demonstrate that the color of the road, dark gray in all point clouds, allows to differentiate it from the sidewalk. This is an interesting finding because the sidewalk color is also gray but less dark. It is important to highlight that for the cases where the training set is composed of point clouds with roads of different types and colors, RGB features will not provide helpful information. This behavior was observed with CARLA data where each town has different types of roads.

Sidewalk: Present in 2 out of 4 scenarios. For both test point clouds, the best performance was obtained in P_4 with RGB features. Results and conclusions are similar to those presented before for road class: RGB features provide helpful information but in the cases where the training set has different types of sidewalks, the color will not help.

Vegetation: Present in all scenarios. For S_0 , the best performance was obtained in P_1 with RGB features. For S_3 , the best performance was obtained in P_1 without RGB features. However, the P_1 scenario did not learn building class so the vegetation can be identified based on their z component. One may observe that the performance in the P_1 scenario is very similar with or without RGB features. In the scenarios where building and vegetation classes were learned (P_2, P_3, P_4), no clear conclusions about the influence of RGB features can be stated. Intuitively, vegetation is green and this color should provide discriminant information. However, the vegetation class has trees with leaves, branches, and trunks of different colors. Additionally, due to registration errors, projected values from color images into point clouds may correspond to *non-vegetation* objects such as the sky in some cases.

Vehicles: Present in all scenarios. For S_0 , the best performance was obtained in P_2 with RGB features. For S_3 , the best performance was obtained in P_1 with RGB features. In the former, RGB features systematically improve the performance by about 10 %. In the latter, no clear influence of the color information. An explanation to this difference of behavior is that S_3 has 5 times more points of vehicle class than S_0 (see Table 2.2). This, together with the presence of more vehicle types, including motorbikes and scooters, demonstrates that color is not a discriminating feature for a class whose objects can have any color.

Traffic-sign: Present in all scenarios. For both test point clouds, the best performance was obtained in P_4 with RGB features. In all other cases, models with color features also obtained the best results. This is expected behavior because traffic signs have some specific colors such as blue,

red, and white. We highlight that this class only contains the objects that provide information about the traffic (stop, parking restrictions) and does not include the vertical pole elements.

Traffic-light: Present in all scenarios. For S_0 , the best performance was obtained in P_2 with RGB features. For S_3 , the best performance was obtained in P_1 without RGB features. In this class, performance is particularly low compared to other classes. In some cases, it is not even learned with or without color features, especially in S_3 . We manually visualized some of the errors in this class and identified that models often confuse traffic lights with vertical poles with intricate designs of Parisian streets.

Static: Present in all scenarios. For both test point clouds, the best performance was obtained in P_2 with RGB features. However, the performance is very low, especially in S_3 . It is an expected result because the class contains many different objects including bus stops, garbage, and stores furniture, where color information is not discriminant.

Terrain: Present in 2 out of 4 scenarios. The class has an inferior performance in all cases. It has a shallow rate in Paris data and the point cloud where it has the highest value is in S_0 . This lack of representativity in the training set prevents learning and results in the test set. This class cannot be identified by using geometrical features because it is the ground where trees are planted and to distinguish it from other ground-like classes, it needs RGB features.

6.7 Discussion

We presented proposed approaches to perform semantic and instance segmentation of urban point clouds. Firstly, we studied two methods based on 2D projections using Semantic KITTI and Street 3D datasets, for semantic segmentation. The first method relies on spherical projections, while the second also includes BEV projections to perform ground detection.

From these results obtained using 2D projections (Section 6.3.2), we systematically found that class imbalance is a common issue that damages performance. We proposed to deal with the most frequent classes using a differential approach to classify ground and other classes separately. This method was implemented during the competition of the SHREC'20 challenge and we obtained better performance than the baseline. However, the use of 2D projections leads to the loss of geometrical information of 3D data.

In the second part of this chapter, we introduced proposed methods to perform semantic segmentation in the novel Paris CARLA 3D dataset using 3D-based architectures. We evaluated two baseline methods for the benchmark and found that, as expected, KPConv architecture performs better than PointNet++ architecture in tested scenarios. The results of performing transfer learning from models initially trained with CARLA point clouds and then with Paris data, demonstrate that the use of synthetic data is an alternative when little real data is available.

The performance in CARLA data is lower than with Paris data with both architectures (see Tables 6.5 and 6.4). It occurs because the variability in synthetic towns is much more bigger than in real world data. The objects present in Paris data in training and test point clouds are more similar than the objects that are in the different towns. Additional, as we described in Section 2.6, class rates and the type of objects present in synthetic towns have strong variations in the dataset.

The issues associated with class imbalance are still present and systematically damage the performance in evaluated scenarios. To solve it, we proposed to handle it by using a hierarchical approach that allows the model to focus on less populated classes. Obtained results demonstrate that the representation of classes through a hierarchical tree improves the performance of less populated classes. In our case, classes were grouped in two depth levels: in the first level, classes are grouped according to their geometrical similarity; in the second level, they are grouped according to the type of use in the context of the autonomous vehicle. Recently, a very similar approach to embed hierarchies in the loss function was proposed by [209]. Our idea was conceived in parallel without knowing about that publication.

In the last part of the chapter (Section 6.5), we present a simple approach to perform instance segmentation tasks in PC3D dataset. The method is based on predicted labels from semantic segmentation and identifies instances based on BEV projections. Obtained results demonstrate that using a hybrid approach using Deep Learning and features extracted using Mathematical Morphology is capable of identifying most of the instances in some cases. Additionally, there is evidence that our novel PC3D dataset is challenging in this type of task due to the presence of moving objects, such as pedestrians and vehicles, that are harder to recognize.

6.8 Conclusions

In this chapter, we present the proposed approaches for semantic segmentation and instance segmentation of 3D point clouds in urban scenes. We compared different alternatives using deep learning models based on 2D projections and directly on 3D data. As a common systematic issue, less populated classes presented lower performance.

Two alternatives to address the class imbalance in semantic segmentation tasks were proposed: use of specialized models for learning different classes; embedding a hierarchy into a loss function. The first approach demonstrates that using a specialized model can provide higher performance for this type of problem; the second approach penalizes the errors differently during learning according to a given hierarchy. In evaluated scenarios, the use of this last approach obtained the highest results in the PC3D dataset.

The last part of the chapter focuses on the analyses of the influence of color information in point clouds segmentation. We compared four learning scenarios to identify when the color provides additional information to better discriminate in some classes. The obtained results demonstrate that color systematically improves models' performance for some classes such as fence, road-line, road, sidewalk, and traffic sign. Moreover, the presence of color did not damage the performance for any class in all scenarios at the same time. It demonstrates that including RGB features, together with a dropout to prevent overfitting, provides better results than using only geometrical features.

Concerning the mean IoU of evaluated scenarios, it is observed that models including color features perform better in all scenarios in S_0 and in two out of four scenarios in S_3 . Additionally, for both point clouds, the highest mean IoU was obtained in the P_4 scenario. It is a very interesting finding because, in this case, all classes were learned. It shows that including together color information and a hierarchical loss performs better than in other scenarios with fewer classes where one may expect they are easier to learn.

Chapter 7

Conclusions and perspectives

7.1 Résumé

Dans ce chapitre, nous présentons les conclusions et perspectives de cette thèse. Nous exposons les principales contributions de chaque chapitre et les résultats les plus importants obtenus au cours de la recherche. Nous proposons également quelques perspectives pour poursuivre l'étude dans ce domaine.

7.2 Conclusions

In this thesis, our goal was to provide automatic tools to improve virtual scenarios and to study the influence of spectral information on the automatic segmentation of urban scenes. We divided our work into multiple stages to study the different aspects of the main objective.

In Chapter 3, we proposed a model to build a representation of color images named Graph-based Color Lines (GCL). It models spectral diversity using a combination of Gaussian Mixture Models (GMM) and a Minimum Spanning Tree. Then, we included spatial information to increase regions compactness of the representation. Our proposal was tested in an over-segmentation task using the BSD500 dataset and compared to SLIC superpixel models using boundary recall, under-segmentation error, compactness, and density. Obtained results demonstrate that GCL can be used as a preprocessing stage to simplify images while preserving spectral diversity.

As a natural extension of GCL, we proposed a model to represent spectral diversity of higher dimensional images such as Multispectral (MSI) or Hyperspectral images (HSI). Similar to GCL for color images, this representation preserves spectral diversity. We tested our model using two different types of multispectral images: a Powders dataset [8] and the University of the Negev [7] dataset. In the first case, we demonstrate that our model preserves spectral information of powders present in images even if they are in a meager quantity. This preservation of spectral diversity using our proposed method is a noteworthy feature for the REPLICA project. For example, traffic signs usually are not as frequent as on the road in urban scenes. However, it is necessary to recognize them, and our proposal does it. In the second case, we evaluated the capability of our proposal to build a representation under uncontrolled lighting conditions, typical of urban scenes. Obtained results were satisfactory using MSI and HSI, demonstrating that the extension of the GCL model,

together with the inclusion of spatial information, is an alternative to build a reliable representation of the images.

In Chapter 4, we present a detailed analysis of the data acquired using the L3D2 platform in the context of the REPLICA project. We evaluated spectral signatures and reflectance histograms for several objects on performed acquisitions. We conclude that the current technology of multispectral cameras cannot acquire reliable images from outdoor scenes. On the one hand, short exposure times in HS cameras lead to shallow reflectance values. On the other hand, more significant exposure times lead to a blurring effect in acquired images. Furthermore, uncontrolled light conditions and moving objects (traffic actors and the scanning system) limit the data quality. As a collateral effect of larger exposure times in HS cameras, a more significant registration error can occur between 2D images and 3D point clouds.

In Chapter 5, we proposed a loss function for semantic segmentation tasks. Loss function has a key role in DL methods because it permits measuring how well predictions during the optimization process in the training stage and, consequently, improves the model performance in the test set. We proposed a new loss function named Power Jaccard for semantic segmentation tasks. It was evaluated with different types of data (grayscale and color images, point cloud projections) and for different applications (digits, urban scenes, plants). Obtained results demonstrate that it outperforms classical losses in evaluated scenarios. Motivated by these good results obtained with images, we tested the power Jaccard in semantic segmentation of 3D point clouds in Chapter 6. Again, the best results were obtained for the models trained using our proposal. We highlight that several data types with different training scenarios were evaluated to demonstrate the validity of the power Jaccard loss.

In Chapter 6, we focused on semantic segmentation of point clouds and studied the influence of class imbalance during learning. We proposed two alternatives to deal with it using a hierarchical approach. The first alternative is based on several models where each one is specialized into a subset of classes from all available ones. For example, model A learns to distinguish *ground-like* classes such as road, sidewalk, and terrain; model B learns to classify *building-like* such as wall and building; model C learns all the other classes. Then, all predictions are put together in order to obtain the prediction for all the possible classes. The second alternative embeds a hierarchical representation of the classes using a tree in the loss function. Our main contribution is the proposal of a loss representing the classes as a hand-crafted tree. This hierarchical representation penalizes differently the types of errors that occurred during learning. Additionally, it permits the students to learn the classes better and improve models' performance. Our proposal permits embedding some *a priori* knowledge that permits the achievement of the highest scores in semantic segmentation of point clouds in evaluated scenarios.

A second contribution related to 3D point clouds segmentation was the novel Paris CARLA 3D (PC3D) dataset proposal. This dataset aims to provide a challenging and voluminous dataset to evaluate and improve existing methods for 3D mapping of outdoor environments on complex computer vision tasks. One of the main goals of the REPLICA project is to increase the realism of simulation platforms for autonomous vehicle. Within this scope, our proposed dataset is composed of virtual and real point clouds from urban scenes. In Chapter 6, we demonstrate that transfer learning can be considered as an alternative to improve models performance for actual data using a pre-training stage with synthetic data. Concerning our contributions for the dataset, we did it in three domains: creating the evaluation protocol of semantic and instance segmentation tasks; the benchmark of semantic and instance segmentation tasks; manual labeling of semantic classes and instances for the ground truth.

A third contribution of Chapter 6 was the study of the influence of color features in point clouds segmentation. We conducted a comparative analysis of different learning scenarios using real-world data from PC3D dataset was presented. A relevant finding was that models including RGB features systematically perform better than models without them in evaluated scenarios. Additionally, we evidenced that the inclusion of color features did not damage the performance for any class in all scenarios at the same time. This last result confirms the interest in including more detailed spectral information available in MSI/HSI to point clouds can help to improve scene description of urban scenes.

7.3 Perspectives

Several directions of research can be studied from results presented in this thesis:

- GCL model can be suitable for applications to enhance color in videos. Recently, [133] proposed a representation using RGB color space in this spirit. Other applications such as color correction and color grading can be considered.
- Current technology of HS cameras does not provide reliable images in outdoor scenes. We expect that in the future, new technology will be able to use short exposure times to prevent the blurring effect of dynamic objects while acquiring good quality images.
- For our proposed power Jaccard loss, we manually selected the value of p in all experiments. In some cases (Tables 5.2 and 5.3, for example), best results were obtained with values different to $p = 2$, where the minimum of derivative is at $\hat{y} = 1$. An interesting perspective is to find a method to automatically learn (or update) the value of p during training.
- We evidenced that the order of providing samples to the model during training affects its performance, especially in highly imbalanced datasets. It is known as *curricula learning*. In the benchmark of our proposed dataset, we used a sphere-based strategy to select samples during training. However, more complex methods using a curriculum to order samples given a *complexity* criteria can improve the performance. For example, a criteria to estimate *complexity* based the number of dimensions to model a class weighted by the inverse of their ratio can be evaluated: *vertical-like* objects (pole and tree trunks) can be fitted with a 1D line and usually are not very frequent; *ground-like* objects (road and sidewalk) with a 2D plane are very often in the most frequent classes.
- Combining classical methods and DL methods provides interesting results. To use some classical methods to learn *easy* classes such as *ground-like* ones, and then learn the rest of the classes using a DL model demonstrated promising results in the SHREC'20 challenge. This strategy, combined with a hierarchical approach using DL models, could be an interesting path.
- The inclusion of a hand-crafted hierarchical tree improved models performance. An alternative to including another type of *a priori* information would be to automatically generate a hierarchical tree based on classes *complexity*, as explained before for curricula learning.
- Our proposed dataset is very challenging for automatic tasks. It is expected that the research community will profit from it to evaluate and improve existing methods for 3D mapping

outdoor environments such as semantic segmentation, instance segmentation, and scene completion.

Bibliography

- [1] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *Computer Vision and Pattern Recognition*, 2016.
- [2] Joseph Tighe and Svetlana Lazebnik. SuperParsing: Scalable Nonparametric Image Parsing with Superpixels. page 14, 2010.
- [3] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011.
- [4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [5] Massimo Minervini, Andreas Fischbach, Hanno Scharr, and Sotirios A Tsaftaris. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern recognition letters*, 81:80–89, 2016.
- [6] Volodymyr Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013.
- [7] Boaz Arad and Ohad Ben-Shahar. Sparse Recovery of Hyperspectral Signal from Natural RGB Images. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, volume 9911, pages 19–34. Springer International Publishing, Cham, 2016.
- [8] Tiancheng Zhi, Bernardo R Pires, Martial Hebert, and Srinivasa G Narasimhan. Multispectral Imaging for Fine-Grained Recognition of Powders on Complex Backgrounds. page 10, 2019.
- [9] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.
- [10] Tao Ku, Remco C Veltkamp, Bas Boom, David Duque-Arias, Santiago Velasco-Forero, Jean-Emmanuel Deschaud, Francois Goulette, Beatriz Marcotegui, Sebastián Ortega, Agustín Trujillo, et al. Shrec 2020: 3d point cloud semantic segmentation for street scenes. *Computers & Graphics*, 93:13–24, 2020.

- [11] Zhongwen Hu, Qin Zou, and Qingquan Li. Watershed superpixel. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 349–353. IEEE, 2015.
- [12] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. TurboPixels: Fast Superpixels Using Geometric Flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, December 2009.
- [13] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC Superpixels, 2010.
- [14] M Schmitt and F Prêteux. Un nouvel algorithme en morphologie mathématique: les rh maxima et rh minima. *Proc. 2ieme Semaine Internationale de l’Image Electronique*, pages 469–475, 1986.
- [15] Simone Kotthaus, Thomas EL Smith, Martin J Wooster, and CSB Grimmond. Derivation of an urban materials spectral library through emittance and reflectance spectroscopy. *ISPRS Journal of Photogrammetry and Remote Sensing*, 94:194–212, 2014.
- [16] Yuan Zhou, Anand Rangarajan, and Paul D Gader. A spatial compositional model for linear unmixing and endmember uncertainty estimation. *IEEE Transactions on Image Processing*, 25(12):5987–6002, 2016.
- [17] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.
- [18] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.
- [19] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. *arXiv:1904.08889 [cs]*, April 2019. arXiv: 1904.08889.
- [20] History of self-driving cars, July 2021. Page Version ID: 1033295521.
- [21] Matthew A Turk, David G. Morgenthaler, Keith D. Gremban, and Martin Marra. Vits-a vision system for autonomous land vehicle navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):342–361, 1988.
- [22] Todd Litman. Implications for Transport Planning. page 46, June 2021.
- [23] D Pomerleau. An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems; Morgan Kaufmann Publishers Inc.: Burlington, MA, USA*, 1998.
- [24] Sergey I. Nikolenko. Introduction: The Data Problem. In Sergey I. Nikolenko, editor, *Synthetic Data for Deep Learning*, Springer Optimization and Its Applications, pages 1–17. Springer International Publishing, Cham, 2021.
- [25] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349, 2016.

- [26] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. 2016.
- [27] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*, 2016.
- [28] Apollo: A platform for autonomous driving.
- [29] Anh Nguyen and Bac Le. 3d point cloud segmentation: A survey. In *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, pages 225–230. IEEE, 2013.
- [30] I. Omer and M. Werman. Color lines: image specific color representation. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages 946–953, Washington, DC, USA, 2004. IEEE.
- [31] D Duque-Arias, Santiago Velasco-Forero, J-E Deschaud, François Goulette, and Beatriz Marcotegui. A graph-based color lines model for image analysis. In *International Conference on Image Analysis and Processing*, pages 181–191. Springer, 2019.
- [32] David Duque-Arias, Santiago Velasco-Forero, Jean-Emmanuel Deschaud, Francois Goulette, Andrés Serna, Etienne Decencière, and Beatriz Marcotegui. On power jaccard losses for semantic segmentation. In *VISAPP 2021: 16th International Conference on Computer Vision Theory and Applications*, 2021.
- [33] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 2017.
- [34] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.
- [35] Lingyan Zhou. M2NIST Segmentation / U-net, 2018.
- [36] J. C. Harsanyi and C. Chang. Hyperspectral image classification and dimensionality reduction: an orthogonal subspace projection approach. *IEEE Transactions on Geoscience and Remote Sensing*, 32(4):779–785, July 1994.
- [37] Gary A Shaw and Hsiaohua K Burke. Spectral imaging for remote sensing. *Lincoln laboratory journal*, 14(1):3–28, 2003.
- [38] M. Govender, K. Chetty, and H. Bulcock. A review of hyperspectral remote sensing and its application in vegetation and water resource studies. *Water SA*, 33(2), January 2007.
- [39] Chein-I Chang. *Hyperspectral data exploitation: theory and applications*. John Wiley & Sons, 2007.
- [40] A. R. Huete. 11 - REMOTE SENSING FOR ENVIRONMENTAL MONITORING. In Janick F. Artiola, Ian L. Pepper, and Mark L. Brusseau, editors, *Environmental Monitoring and Characterization*, pages 183–206. Academic Press, Burlington, January 2004.

- [41] Marco J. Giardino. A history of NASA remote sensing contributions to archaeology. *Journal of Archaeological Science*, 38(9):2003–2009, September 2011.
- [42] Freek D. van der Meer, Harald M. A. van der Werff, Frank J. A. van Ruitenbeek, Chris A. Hecker, Wim H. Bakker, Marleen F. Noomen, Mark van der Meijde, E. John M. Carranza, J. Boudewijn de Smeth, and Tsehaie Woldai. Multi- and hyperspectral geologic remote sensing: A review. *International Journal of Applied Earth Observation and Geoinformation*, 14(1):112–128, February 2012.
- [43] Telmo Adão, Jonáš Hruška, Luís Pádua, José Bessa, Emanuel Peres, Raul Morais, and Joaquim João Sousa. Hyperspectral Imaging: A Review on UAV-Based Sensors, Data Processing and Applications for Agriculture and Forestry. *Remote Sensing*, 9(11):1110, November 2017.
- [44] Elhadi Adam, Onesimo Mutanga, and Denis Rugege. Multispectral and hyperspectral remote sensing for identification and mapping of wetland vegetation: a review. *Wetlands Ecology and Management*, 18(3):281–296, June 2010.
- [45] Cong Phuoc Huynh, Samir Mustapha, Peter Runcie, and Fatih Porikli. Multi-class support vector machines for paint condition assessment on the Sydney Harbour Bridge using hyperspectral imaging. *Structural Monitoring and Maintenance*, 2(3):181–197, September 2015.
- [46] Yuan-Yuan Pu, Yao-Ze Feng, and Da-Wen Sun. Recent Progress of Hyperspectral Imaging on Quality and Safety Inspection of Fruits and Vegetables: A Review. *Comprehensive Reviews in Food Science and Food Safety*, 14(2):176–188, 2015.
- [47] Rafik Naccache, Anna Mazhorova, Matteo Clerici, Riccardo Piccoli, Larousse Khosravi Khorashad, Alexander O. Govorov, Luca Razzari, Fiorenzo Vetrone, and Roberto Morandotti. Terahertz Thermometry: Combining Hyperspectral Imaging and Temperature Mapping at Terahertz Frequencies. *Laser & Photonics Reviews*, 11(5):1600342, 2017.
- [48] Anguo Xie, Da-Wen Sun, Zhiwei Zhu, and Hongbin Pu. Nondestructive Measurements of Freezing Parameters of Frozen Porcine Meat by NIR Hyperspectral Imaging. *Food and Bioprocess Technology*, 9(9):1444–1454, September 2016.
- [49] Qian Yang, Da-Wen Sun, and Weiwei Cheng. Development of simplified models for nondestructive hyperspectral imaging monitoring of TVB-N contents in cured meat during drying process. *Journal of Food Engineering*, 192:53–60, January 2017.
- [50] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pages 1–4. IEEE, 2011.
- [51] Qian Wang and Min-Koo Kim. Applications of 3d point cloud data in the construction industry: A fifteen-year review from 2004 to 2018. *Advanced Engineering Informatics*, 39:306–319, 2019.
- [52] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

- [53] Adam Hoover, Gillian Jean-Baptiste, Xiaoyi Jiang, Patrick J Flynn, Horst Bunke, Dmitry B Goldgof, Kevin Bowyer, David W Eggert, Andrew Fitzgibbon, and Robert B Fisher. An experimental comparison of range image segmentation algorithms. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):673–689, 1996.
- [54] Andrés Serna and Beatriz Marcotegui. Detection, segmentation and classification of 3d urban objects using mathematical morphology and supervised learning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 93:243–255, July 2014.
- [55] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.
- [56] Eren Erdal Aksoy, Saimir Baci, and Selcuk Cavdar. Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 926–932. IEEE, 2019.
- [57] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.
- [58] Yuan Wang, Tianyue Shi, Peng Yun, Lei Tai, and Ming Liu. Pointseg: Real-time semantic segmentation based on 3d lidar point cloud. *arXiv preprint arXiv:1807.06288*, 2018.
- [59] Dan Claudiu Cireşan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *Twenty-second international joint conference on artificial intelligence*, 2011.
- [60] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. Ieee, 2017.
- [61] Victoria Plaza-Leiva, Jose Antonio Gomez-Ruiz, Anthony Mandow, and Alfonso García-Cerezo. Voxel-based neighborhood for spatial shape pattern classification of lidar point clouds with supervised learning. *Sensors*, 17(3):594, 2017.
- [62] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Classification of point cloud scenes with multiscale voxel deep network. *arXiv preprint arXiv:1804.03583*, 2018.
- [63] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.
- [64] Yusheng Xu, Xiaohua Tong, and Uwe Stilla. Voxel-based representation of 3d point clouds: Methods, applications, and its potential use in the construction industry. *Automation in Construction*, 126:103675, 2021.
- [65] Saifullahi Aminu Bello, Shangshu Yu, Cheng Wang, Jibril Muhmmad Adam, and Jonathan Li. deep learning on 3d point clouds. *Remote Sensing*, 12(11):1729, 2020.

- [66] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018.
- [67] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1607–1616, 2019.
- [68] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y Zeevi. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315, 1997.
- [69] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019.
- [70] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11632–11641, 2020.
- [71] Hugues Thomas, François Goulette, Jean-Emmanuel Deschaud, Beatriz Marcotegui, and Yann LeGall. Semantic classification of 3d point clouds with multiscale spherical neighborhoods. In *2018 International conference on 3D vision (3DV)*, pages 390–398. IEEE, 2018.
- [72] Timo Hackel, Jan D Wegner, and Konrad Schindler. Fast semantic segmentation of 3d point clouds with strongly varying density. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, 3:177–184, 2016.
- [73] I Puente, H González-Jorge, J Martínez-Sánchez, and P Arias. Review of mobile mapping and surveying technologies. *Measurement*, 46(7):2127–2145, 2013.
- [74] Cheng Wang, Chenglu Wen, Yudi Dai, Shangshu Yu, and Minghao Liu. Urban 3d modeling with mobile laser scanning: a review. *Virtual Reality & Intelligent Hardware*, 2(3):175–212, 2020.
- [75] F Goulette, F Nashashibi, I Abuhadrous, S Ammoun, and C Lourceau. An integrated on-board laser range sensing system for on-the-way city and road modelling. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(A), 2006.
- [76] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Paris-lille-3d: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. *The International Journal of Robotics Research*, 37(6):545–557, 2018.
- [77] Velodyne’s HDL-32E Surround Lidar Sensor.
- [78] Andreas Nuchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6d slam—3d mapping outdoor environments. *Journal of Field Robotics*, 24(8-9):699–722, 2007.
- [79] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

- [80] Jean-Emmanuel Deschaud. Imls-slam: scan-to-model matching based on 3d data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2480–2485. IEEE, 2018.
- [81] Derek D Lichti, David Jarron, Wynand Tredoux, Mozhddeh Shahbazi, and Robert Radovanovic. Geometric modelling and calibration of a spherical camera imaging system. *The Photogrammetric Record*, 35(170):123–142, 2020.
- [82] Darren J Turner, Zbyněk Malenovský, Arko Lucieer, Johanna D Turnbull, and Sharon A Robinson. Optimizing spectral and spatial resolutions of unmanned aerial system imaging sensors for monitoring antarctic vegetation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(10):3813–3825, 2019.
- [83] Bethany Melville, Arko Lucieer, and Jagannath Aryal. Classification of lowland native grassland communities using hyperspectral unmanned aircraft system (uas) imagery in the tasmanian midlands. *Drones*, 3(1):5, 2019.
- [84] Pedro J Navarro, Leanne Miller, Alberto Gila-Navarro, María Victoria Díaz-Galián, Diego J Aguila, and Marcos Egea-Cortines. 3deepm: An ad hoc architecture based on deep learning methods for multispectral image classification. *Remote Sensing*, 13(4):729, 2021.
- [85] Gunnar Ritt and Bernd Eberle. Use of complementary wavelength bands for laser dazzle protection. In *Technologies for Optical Countermeasures XVI*, volume 11161, page 1116109. International Society for Optics and Photonics, 2019.
- [86] Helge Aasen, Eija Honkavaara, Arko Lucieer, and Pablo J Zarco-Tejada. Quantitative remote sensing at ultra-high resolution with uav spectroscopy: a review of sensor technology, measurement procedures, and data correction workflows. *Remote Sensing*, 10(7):1091, 2018.
- [87] Photonfocus AG. User Manual - Hyperspectral Series MAN068 10/2019 V1.3, October 2019.
- [88] Sofiane Mihoubi, Olivier Losson, Benjamin Mathon, and Ludovic Macaire. Multispectral demosaicing using pseudo-panchromatic image. *IEEE Transactions on Computational Imaging*, 3(4):982–995, 2017.
- [89] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.
- [90] Tomohiro Nishikawa and Yuichi Tanaka. Dynamic Color Lines. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2247–2251, Athens, October 2018. IEEE.
- [91] Xiaoming Yu, Ge Li, Zhenqiang Ying, and Xiaoqiang Guo. A New Shadow Removal Method Using Color-Lines. pages 307–319, 2017.
- [92] Raanan Fattal. Dehazing Using Color-Lines. In *ACM Transactions on Graphics*, volume 34, pages 1–14, December 2014.
- [93] Theo Gevers and Arnold W M Smeulders. Color-based object recognition. *Pattern Recognition*, page 12, 1999.

- [94] Konstantinos N Plataniotis and Anastasios N Venetsanopoulos. *Color image processing and applications*. Springer Science & Business Media, 2013.
- [95] Yining Deng, B. S. Manjunath, C. Kenney, M. S. Moore, and H. Shin. An efficient color representation for image retrieval. *IEEE Transactions on Image Processing*, 10(1):140–147, January 2001.
- [96] Xiang-Yang Wang, Ting Wang, and Juan Bu. Color image segmentation using pixel wise support vector machine classification. *Pattern Recognition*, 44(4):777–787, April 2011.
- [97] Haim Permuter, Joseph Francos, and Ian Jermyn. A study of Gaussian mixture models of color and texture features for image classification and segmentation. *Pattern Recognition*, 39(4):695–706, April 2006.
- [98] Daniel Cremers, Mikael Rousson, and Rachid Deriche. A Review of Statistical Approaches to Level Set Segmentation: Integrating Color, Texture, Motion and Shape. *International Journal of Computer Vision*, 72(2):195–215, April 2007.
- [99] Michèle Gouiffès and Bertrand Zavidovique. Body color sets: A compact and reliable representation of images. *Journal of Visual Communication and Image Representation*, 22(1):48–60, January 2011.
- [100] Yağiz Aksoy, Tunç Ozan Aydın, Aljoša Smolić, and Marc Pollefeys. Unmixing-Based Soft Color Segmentation for Image Manipulation. *ACM Trans. Graph.*, 36(2), March 2017.
- [101] Jesús Angulo. Morphological colour operators in totally ordered lattices based on distances: Application to image filtering, enhancement and analysis. *Computer Vision and Image Understanding*, 107(1-2):56–73, July 2007.
- [102] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018.
- [103] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8092–8101, 2019.
- [104] Nikhil R Pal and Sankar K Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, September 1993.
- [105] H. D. Cheng, X. H. Jiang, Y. Sun, and Jingli Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12):2259–2281, December 2001.
- [106] Y. Ohta, Takeo Kanade, and T. Sakai. Color Information for Region Segmentation. *Computer Graphics and Image Processing*, 13(3):222–241, July 1980.
- [107] Ron Ohlander, Keith Price, and D. Raj Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8(3):313–333, December 1978.

- [108] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011.
- [109] Ren and Malik. Learning a classification model for segmentation. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 10–17 vol.1, October 2003.
- [110] Dana E. Ilea and Paul F. Whelan. Image segmentation based on the integration of colour–texture descriptors—A review. *Pattern Recognition*, 44(10):2479–2501, October 2011.
- [111] Thomas Brox, Mikael Rousson, Rachid Deriche, and Joachim Weickert. Unsupervised segmentation incorporating colour, texture, and motion. In *International conference on computer analysis of images and patterns*, pages 353–360, 2003. bibtex*[organization=Springer].
- [112] Chung-Lin Huang, Tai-Yuen Cheng, and Chaur-Chin Chen. Color images segmentation using scale space filter and Markov random field. *Pattern Recognition*, 25(10):1217–1229, 1992. bibtex*[publisher=Elsevier].
- [113] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, 2018.
- [114] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *Computer Vision, IEEE International Conference on*, volume 2, pages 10–10. IEEE Computer Society, 2003.
- [115] V. Machairas, E. Decenci ere, and T. Walter. Waterpixels: Superpixels based on the watershed transformation. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 4343–4347, October 2014.
- [116] Vaia Machairas, Matthieu Faessel, David C ardenas-Pe na, Th odore Chabardes, Thomas Walter, and Etienne Decenci ere. Waterpixels. *IEEE Transactions on Image Processing*, 24(11):3707–3716, 2015.
- [117] Serge Beucher and Fernand Meyer. The morphological approach to segmentation: the watershed transformation. *Optical Engineering-New York-Marcel Dekker Incorporated-*, 34:433–433, 1992. bibtex*[publisher=Marcel Dekker AG].
- [118] Abdelhameed Ibrahim and El-Sayed M El-kenawy. Applications and datasets for superpixel techniques: A survey. *Journal of Computer Science and Information Systems*, 15(3), 2020.
- [119] Umme Sara, Morium Akter, and Mohammad Shorif Uddin. Image quality assessment through fsim, ssim, mse and psnr—a comparative study. *Journal of Computer and Communications*, 7(3):8–18, 2019.
- [120] Jacob S ogaard, Luk as Krasula, Muhammad Shahid, Dogancan Temel, Kjell Brunnstr om, and Manzoor Razaak. Applicability of existing objective metrics of perceptual quality for adaptive video streaming. *Electronic Imaging*, 2016(13):1–7, 2016.
- [121] Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin de Capitani, and Luc Van Gool. Seeds: Superpixels extracted via energy-driven sampling. In *European conference on computer vision*, pages 13–26. Springer, 2012.

- [122] Peer Neubert and Peter Protzel. Superpixel benchmark and comparison. In *Proc. Forum Bildverarbeitung*, volume 6, pages 1–12, 2012.
- [123] Alexander Schick, Mika Fischer, and Rainer Stiefelhagen. Measuring and evaluating the compactness of superpixels. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 930–934. IEEE, 2012.
- [124] Antoni Buades, Jose Luis Lisani, and Jean-Michel Morel. On the distribution of colors in natural images. page 14, 2010.
- [125] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.
- [126] Sreevani and C.A. Murthy. On bandwidth selection using minimal spanning tree for kernel density estimation. *Computational Statistics & Data Analysis*, 102:67 – 84, 2016.
- [127] Zhiding Yu, Oscar C Au, Ketan Tang, and Chunjing Xu. Nonparametric density estimation on a graph: Learning framework, fast approximation and application in image segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2201–2208. IEEE, 2011.
- [128] A. M. Martinez and A. C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, February 2001.
- [129] Jiali Cui, Yunhong Wang, JunZhou Huang, Tieniu Tan, and Zhenan Sun. An iris image synthesis method based on PCA and super-resolution. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 4, pages 471–474 Vol.4, August 2004.
- [130] James Burger and Aoife Gowen. Data handling in hyperspectral image analysis. *Chemometrics and Intelligent Laboratory Systems*, 108(1):13–22, 2011.
- [131] Gustavo Camps-Valls, Tatyana V Bandos Marshева, and Dengyong Zhou. Semi-supervised graph-based hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 45(10):3044–3054, 2007.
- [132] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [133] Zheng-Jun Du, Kai-Xiang Lei, Kun Xu, Jianchao Tan, and Yotam Gingold. Video recoloring via spatial-temporal geometric palettes. *ACM Transactions on Graphics (TOG)*, 40(4):1–16, 2021.
- [134] Yannis Ioannidis. The history of histograms (abridged). In *Proceedings 2003 VLDB Conference*, pages 19–30. Elsevier, 2003.
- [135] Li Deng and Dong Yu. Deep learning: methods and applications. *Foundations and trends in signal processing*, 7(3–4):197–387, 2014.

- [136] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.
- [137] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [138] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [139] Xue Ying. An overview of overfitting and its solutions. In *Journal of Physics: Conference Series*, volume 1168, page 022022. IOP Publishing, 2019.
- [140] Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Are loss functions all the same? *Neural computation*, 16(5):1063–1076, 2004.
- [141] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, 2019.
- [142] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.
- [143] Ma Jun. Segmentation loss odyssey. *arXiv preprint arXiv:2005.13449*, 2020.
- [144] Francesco Calivá, Claudia Iriondo, Alejandro Morales Martinez, Sharmila Majumdar, and Valentina Pedoia. Distance map loss penalty term for semantic segmentation. *arXiv preprint arXiv:1908.03679*, 2019.
- [145] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Bridging category-level and instance-level semantic image segmentation. *arXiv preprint arXiv:1605.06885*, 2016.
- [146] Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour. Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *International Workshop on Machine Learning in Medical Imaging*, pages 379–387. Springer, 2017.
- [147] Mark Polak, Hong Zhang, and Minghong Pi. An evaluation metric for image segmentation of multiple objects. *Image and Vision Computing*, 27(8):1223–1227, 2009.
- [148] Hoel Kervadec, Jihene Bouchtiba, Christian Desrosiers, Eric Granger, Jose Dolz, and Ismail Ben Ayed. Boundary loss for highly unbalanced segmentation. *arXiv preprint arXiv:1812.07032*, 2018.
- [149] Su Yang, Jihoon Kweon, and Young-Hak Kim. Major vessel segmentation on x-ray coronary angiography using deep networks with a novel penalty loss function. In *International Conference on Medical Imaging with Deep Learning—Extended Abstract Track*, 2019.
- [150] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pages 240–248. Springer, 2017.

- [151] Paul Jaccard. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. *Bull Soc Vaudoise Sci Nat*, 37:241–272, 1901.
- [152] Michel Marie Deza and Elena Deza. Encyclopedia of distances. In *Encyclopedia of distances*, pages 1–583. Springer, 2009.
- [153] Flavio Chierichetti, Ravi Kumar, Sandeep Pandey, and Sergei Vassilvitskii. Finding the jaccard median. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 293–311. SIAM, 2010.
- [154] Md Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International symposium on visual computing*, pages 234–244. Springer, 2016.
- [155] I. Martire, P.N. da Silva, Alexandre Plastino, Fabio Fabris, and Alex A. Freitas. A novel probabilistic jaccard distance measure for classification of sparse and uncertain data. In Elaine Rebeiro de Faria Paiva, Luiz Merschmann, and Ricardo Cerri, editors, *5th Brazilian Symposium on Knowledge Discovery, Mining and Learning (KDMiLe)*, pages 81–88, October 2017.
- [156] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.
- [157] Etienne Decencière, Santiago Velasco-Forero, Fu Min, Juanjuan Chen, H el ene Burdin, Gervais Gauthier, Bruno La y, Thomas Borschloegl, and Th er ese Baldeweck. Dealing with topological information within a fully convolutional neural network. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 462–471. Springer, 2018.
- [158] Peter E Hart, David G Stork, and Richard O Duda. *Pattern classification*. Wiley Hoboken, 2000.
- [159] Simone Scardapane and Dianhui Wang. Randomness in neural networks: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(2):e1200, 2017.
- [160] Jorge Hern andez and Beatriz Marcotegui. Point cloud segmentation towards urban ground modeling. In *2009 Joint Urban Remote Sensing Event*, pages 1–5. IEEE, 2009.
- [161] Matthew D Zeiler. Adadelata: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
- [162] Ethem Alpaydin. Voting over multiple condensed nearest neighbors. In *Lazy learning*, pages 115–132. Springer, 1997.
- [163] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Towards semantic segmentation of urban-scale 3d point clouds: A dataset, benchmarks and challenges. *arXiv preprint arXiv:2009.03137*, 2020.
- [164] Bir Bhanu, Sungkee Lee, Chih-Cheng Ho, and Tom Henderson. Range data processing: Representation of surfaces by edges. In *Proceedings of the eighth international conference on pattern recognition*, pages 236–238. IEEE Computer Society Press, 1986.

- [165] Angel Domingo Sappa and Michel Devy. Fast range image segmentation by an edge detection strategy. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 292–299. IEEE, 2001.
- [166] Andrés Serna. *Semantic analysis of 3D point clouds from urban environments: ground, facades, urban objects and accessibility*. PhD thesis, MINES ParsTech, 2014.
- [167] Bruno Vallet, Mathieu Brédif, Andrés Serna, Beatriz Marcotegui, and Nicolas Papanicolaou. Terramobilita/iqmulus urban point cloud analysis benchmark. *Computers & Graphics*, 49:126–133, 2015.
- [168] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.
- [169] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5010–5019, 2018.
- [170] Biao Leng, Shuang Guo, Xiangyang Zhang, and Zhang Xiong. 3d object retrieval with stacked local convolutional autoencoder. *Signal Processing*, 112:119–128, 2015.
- [171] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.
- [172] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.
- [173] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.
- [174] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.
- [175] Yosuke Nakagawa, Hideaki Uchiyama, Hajime Nagahara, and Rin-Ichiro Taniguchi. Estimating surface normals with depth image gradients for fast and accurate registration. In *2015 International Conference on 3D Vision*, pages 640–647. IEEE, 2015.
- [176] Jérôme Demantké, Clément Mallet, Nicolas David, and Bruno Vallet. Dimensionality based scale selection in 3d lidar point clouds. In *Laserscanning*, 2011.
- [177] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.
- [178] C. Lantuejoul and F. Maisonneuve. Geodesic methods in quantitative image analysis. *Pattern Recognition*, 17(2):177 – 187, 1984.

- [179] Vincent Morard, Etienne Decenciere, and Petr Dokládál. Efficient geodesic attribute thinnings based on the barycentric diameter. *Journal of mathematical imaging and vision*, 46(1):128–142, 2013.
- [180] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020.
- [181] Thomas Chaton, Nicolas Chaulet, Sofiane Horache, and Loic Landrieu. Torch-points3d: A modular multi-task framework for reproducible deep learning on 3d point clouds. *arXiv preprint arXiv:2010.04642*, 2020.
- [182] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004.
- [183] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.
- [184] Nidhi Ranjan, Pranav Vinod Machingal, Sunil Sri Datta Jammalmadka, Veena Thenaknidiyoor, and AD Dileep. Hierarchical approach for breast cancer histopathology images classification. 2018.
- [185] Somayah Albaradei, Yang Wang, Liangliang Cao, and Li-Jia Li. Learning mid-level features from object hierarchy for image classification. In *IEEE Winter Conference on Applications of Computer Vision*, pages 235–240. IEEE, 2014.
- [186] Zeya Wang, Nanqing Dong, Wei Dai, Sean D Rosario, and Eric P Xing. Classification of breast cancer histopathological images using convolutional neural networks with hierarchical loss and global pooling. In *International Conference Image Analysis and Recognition*, pages 745–753. Springer, 2018.
- [187] Bruce R Muller and William AP Smith. A hierarchical loss for semantic segmentation. In *VISIGRAPP (4: VISAPP)*, pages 260–267, 2020.
- [188] Zhaoyang Niu, Guoqiang Zhong, and Hui Yu. A review on the attention mechanism of deep learning. *Neurocomputing*, 452:48–62, 2021.
- [189] Duarte Fernandes, António Silva, Rafael Névoa, Cláudia Simões, Dibet Gonzalez, Miguel Guevara, Paulo Novais, João Monteiro, and Pedro Melo-Pinto. Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy. *Information Fusion*, 68:161–191, 2021.
- [190] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*, 2016.
- [191] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

- [192] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018.
- [193] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018.
- [194] Gregory P Meyer, Jake Charland, Darshan Hegde, Ankit Laddha, and Carlos Vallespi-Gonzalez. Sensor fusion for joint 3d object detection and semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- [195] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018.
- [196] Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. *arXiv preprint arXiv:1903.01864*, 2019.
- [197] Xin Zhao, Zhe Liu, Ruolan Hu, and Kaiqi Huang. 3d object detection using scale invariant and feature reweighting networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9267–9274, 2019.
- [198] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.
- [199] Hongwu Kuang, Bei Wang, Jianping An, Ming Zhang, and Zehan Zhang. Voxel-fpn: Multi-scale voxel feature aggregation for 3d object detection from lidar point clouds. *Sensors*, 20(3):704, 2020.
- [200] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.
- [201] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Ipod: Intensive point-based object detector for point cloud. *arXiv preprint arXiv:1812.05276*, 2018.
- [202] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9775–9784, 2019.
- [203] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4612, 2020.
- [204] Stefano Gasperini, Mohammad-Ali Nikouei Mahani, Alvaro Marcos-Ramiro, Nassir Navab, and Federico Tombari. Panoster: End-to-end panoptic segmentation of lidar point clouds. *IEEE Robotics and Automation Letters*, 6(2):3216–3223, 2021.

- [205] Zixiang Zhou, Yang Zhang, and Hassan Foroosh. Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation. *arXiv preprint arXiv:2103.14962*, 2021.
- [206] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-deeplab. *arXiv preprint arXiv:1910.04751*, 2019.
- [207] Quang-Hieu Pham, Thanh Nguyen, Binh-Son Hua, Gemma Roig, and Sai-Kit Yeung. Jsis3d: Joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2019.
- [208] Andres Milioto, Jens Behley, Chris McCool, and Cyrill Stachniss. Lidar panoptic segmentation for autonomous driving. IROS, 2020.
- [209] Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12506–12515, 2020.

RÉSUMÉ

Les nuages de points ont suscité l'intérêt de la communauté de recherche au cours des dernières années. Au départ, ils étaient surtout utilisés pour des applications de télédétection. Plus récemment, grâce au développement de capteurs à faible coût et à la publication de plusieurs bibliothèques open source, ils sont devenus très populaires et ont été appliqués à un plus grand nombre d'applications. L'une d'entre elles est la voiture autonome, pour lequel de nombreux efforts ont été déployés au cours du siècle dernier pour le rendre réel. L'évaluation des algorithmes proposés constitue aujourd'hui un goulot d'étranglement très important pour la voiture autonome. En raison du grand nombre de scénarios possibles, il n'est pas possible de l'effectuer dans la vie réelle. Une alternative consiste à simuler des environnements virtuels où toutes les configurations possibles peuvent être établies à l'avance. Cependant, ces environnements ne sont pas aussi réalistes que le monde réel. Dans cette thèse, nous avons étudié la pertinence d'inclure des images hyperspectrales dans la création de nouveaux environnements virtuels. De plus, nous avons proposé de nouvelles méthodes pour améliorer la compréhension des scènes 3D pour la voiture autonome.

MOTS CLÉS

Nuages de points, apprentissage profond, fonctions de perte, représentation d'images, segmentation sémantique, images hyperspectrales, traitement d'images.

ABSTRACT

Point clouds have attracted the interest of the research community over the last years. Initially, they were mostly used for remote sensing applications. More recently, thanks to the development of low-cost sensors and the publication of some open source libraries, they have become very popular and have been applied to a wider range of applications. One of them is the autonomous vehicle where many efforts have been made in the last century to make it real. A very important bottleneck nowadays for the autonomous vehicle is the evaluation of the proposed algorithms. Due to the huge number of possible scenarios, it is not feasible to perform it in real life. An alternative is to simulate virtual environments where all possible configurations can be set up beforehand. However, they are not as realistic as the real world is. In this thesis, we studied the pertinence of including hyperspectral images in the creation of new virtual environments. Furthermore, we proposed new methods to improve 3D scene understanding for autonomous vehicles.

KEYWORDS

Point clouds, Deep Learning, loss functions, image representation, semantic segmentation, hyperspectral images, Image processing.