



Pedro Miguel Freire Faustino

Licenciado em Engenharia Eletrotécnica e de Computadores

Vegetation Detection and Classification for Power Line Monitoring

Dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Engenharia Electrotécnica e de Computadores

Adviser: José António Barata de Oliveira, Associate Professor,
NOVA University of Lisbon

Co-adviser: Francisco Marques, Research Engineer, UNINOVA-CTS

Examination Committee

Chair: Associate Prof. Fernando José Almeida Vieira do Coito

Rapporteur: Assistant Prof. João Almeida das Rosas



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

January, 2022

Vegetation Detection and Classification for Power Line Monitoring

Copyright © Pedro Miguel Freire Faustino, Faculty of Sciences and Technology, NOVA University Lisbon.

The Faculty of Sciences and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

ACKNOWLEDGEMENTS

I wish to express my gratitude to my adviser, Prof. José António Barata de Oliveira, for the opportunity to work under his guidance in such an interesting project. Furthermore, I would like to show my appreciation for his passion for Robotics and how his enthusiasm carries over to his students.

I would also like to thank Francisco Marques for his continued support throughout this work, providing crucial feedback and helpful advice during both the implementation of this work and writing of this document.

I am thankful to Universidade Nova de Lisboa and to all its associates for their commitment to raise great professionals. To all the professors for sharing their invaluable knowledge and their dedication for their respective study areas.

A special thanks to all my classmates and friends, within or outside this academic institution, with who I shared a good portion of my free time doing other leisure activities.

I would like to dedicate this dissertation to my family with a special mention to my parents, Paulo and Margarida, and my brother, André. Their unconditional love and support, throughout my entire life, gives me the strength to always thrive and want to reach greater achievements.

ABSTRACT

Electrical network maintenance inspections must be regularly executed, to provide a continuous distribution of electricity. In forested countries, the electrical network is mostly located within the forest. For this reason, during these inspections, it is also necessary to assure that vegetation growing close to the power line does not potentially endanger it, provoking forest fires or power outages.

Several remote sensing techniques have been studied in the last years to replace the labor-intensive and costly traditional approaches, be it field based or airborne surveillance. Besides the previously mentioned disadvantages, these approaches are also prone to error, since they are dependent of a human operator's interpretation. In recent years, Unmanned Aerial Vehicle (UAV) platform applicability for this purpose has been under debate, due to its flexibility and potential for customisation, as well as the fact it can fly close to the power lines.

The present study proposes a vegetation management and power line monitoring method, using a UAV platform. This method starts with the collection of point cloud data in a forest environment composed of power line structures and vegetation growing close to it. Following this process, multiple steps are taken, including: detection of objects in the working environment; classification of said objects into their respective class labels using a feature-based classifier, either vegetation or power line structures; optimisation of the classification results using point cloud filtering or segmentation algorithms. The method is tested using both synthetic and real data of forested areas containing power line structures. The Overall Accuracy of the classification process is about 87% and 97-99% for synthetic and real data, respectively. After the optimisation process, these values were refined to 92% for synthetic data and nearly 100% for real data. A detailed comparison and discussion of results is presented, providing the most important evaluation metrics and a visual representations of the attained results.

Keywords: Vegetation Management, Power Line Mapping, Remote Sensing, UAV, Point Cloud Classification, Point Cloud Filtering, Point Cloud Segmentation

RESUMO

Manutenções regulares da rede elétrica devem ser realizadas de forma a assegurar uma distribuição contínua de eletricidade. Em países com elevada densidade florestal, a rede elétrica encontra-se localizada maioritariamente no interior das florestas. Por isso, durante estas inspeções, é necessário assegurar também que a vegetação próxima da rede elétrica não a coloca em risco, provocando incêndios ou falhas elétricas.

Diversas técnicas de deteção remota foram estudadas nos últimos anos para substituir as tradicionais abordagens dispendiosas com mão-de-obra intensiva, sejam elas através de vigilância terrestre ou aérea. Além das desvantagens mencionadas anteriormente, estas abordagens estão também sujeitas a erros, pois estão dependentes da interpretação de um operador humano. Recentemente, a aplicabilidade de plataformas com *Unmanned Aerial Vehicles (UAV)* tem sido debatida, devido à sua flexibilidade e potencial personalização, assim como o facto de conseguirem voar mais próximas das linhas elétricas.

O presente estudo propõe um método para a gestão da vegetação e monitorização da rede elétrica, utilizando uma plataforma UAV. Este método começa pela recolha de dados *point cloud* num ambiente florestal composto por estruturas da rede elétrica e vegetação em crescimento próximo da mesma. Em seguida, múltiplos passos são seguidos, incluindo: deteção de objetos no ambiente; classificação destes objetos com as respetivas etiquetas de classe através de um classificador baseado em *features*, vegetação ou estruturas da rede elétrica; otimização dos resultados da classificação utilizando algoritmos de filtragem ou segmentação de *point cloud*. Este método é testado usando dados sintéticos e reais de áreas florestais com estruturas elétricas. A exatidão do processo de classificação é cerca de 87% e 97-99% para os dados sintéticos e reais, respetivamente. Após o processo de otimização, estes valores aumentam para 92% para os dados sintéticos e cerca de 100% para os dados reais. Uma comparação e discussão de resultados é apresentada, fornecendo as métricas de avaliação mais importantes e uma representação visual dos resultados obtidos.

Palavras-chave: Gestão de Vegetação, Mapeamento da Rede Elétrica, Deteção Remota, UAV, Classificação de *Point Cloud*, Segmentação de *Point Cloud*, Filtragem de *Point Cloud*

CONTENTS

| | |
|---|-------------|
| List of Figures | xiii |
| List of Tables | xv |
| Acronyms | xix |
| 1 Introduction | 1 |
| 1.1 Scope | 1 |
| 1.2 Problem | 2 |
| 1.3 Motivation | 2 |
| 1.4 Solution | 3 |
| 1.5 Proposed Method | 3 |
| 1.6 Document Organisation | 3 |
| 2 Related Work | 5 |
| 2.1 Data Collection | 5 |
| 2.1.1 Laser Scanning | 6 |
| 2.1.2 Computer Vision | 7 |
| 2.1.3 Comparison of Approaches | 9 |
| 2.2 Data Analysis | 12 |
| 2.2.1 Synthetic Aperture Radar (SAR) Images | 12 |
| 2.2.2 Optical Satellite Images | 14 |
| 2.2.3 Optical Aerial Images | 15 |
| 2.2.4 Thermal Images | 17 |
| 2.2.5 Mobile Laser Scanning (MLS) | 18 |
| 2.2.6 Airborne Laser Scanning (ALS) | 20 |
| 2.2.7 Unmanned Aerial Vehicle (UAV) | 25 |
| 2.2.8 Comparison of Approaches | 27 |
| 3 Method | 29 |
| 3.1 Concept Overview | 30 |
| 3.2 Proposed Workflow | 32 |
| 3.3 External Resources | 34 |

CONTENTS

| | | |
|----------|---|-----------|
| 3.3.1 | Robot Operating System (ROS) | 34 |
| 3.3.2 | Point Cloud Library (PCL) | 35 |
| 3.3.3 | Aerial Informatics and Robotics Simulation (AirSim) | 35 |
| 3.3.4 | Octomap | 35 |
| 3.3.5 | MeshLab | 35 |
| 3.3.6 | Computational Geometry Algorithms Library (CGAL) | 36 |
| 3.4 | Data Acquisition | 36 |
| 3.4.1 | Synthetic Data | 37 |
| 3.4.2 | Real Data | 41 |
| 3.4.3 | Overview | 42 |
| 3.5 | Point Cloud Classification | 43 |
| 3.5.1 | Input Data Structures | 43 |
| 3.5.2 | Feature Analysis | 44 |
| 3.5.3 | Classifier | 47 |
| 3.5.4 | Classification Functions | 49 |
| 3.5.5 | Evaluation | 50 |
| 3.6 | Post-Processing | 51 |
| 3.6.1 | Filtering | 52 |
| 3.6.2 | Segmentation | 53 |
| 4 | Results | 59 |
| 4.1 | Classification | 59 |
| 4.2 | Optimisation | 63 |
| 5 | Conclusions and Future Work | 69 |
| 5.1 | Conclusions | 69 |
| 5.2 | Future Work | 72 |
| | Bibliography | 75 |

LIST OF FIGURES

| | | |
|-----|---|----|
| 3.1 | Diagrammatic representation of the Proposed Workflow. | 32 |
| 3.2 | Sequence diagram of Synthetic Data Acquisition process. | 38 |
| 3.3 | Screenshot of the Synthetic Data simulated environment. | 39 |
| 3.4 | Screenshot of the resulting Point Cloud Data (PCD) file after mapping the simulated environment. | 41 |
| 3.5 | Close-ups of the real life LiDAR sample Dataset from Open.NRW. | 42 |
| 4.1 | Ground Truth Subset from sample real life Dataset. | 60 |
| 4.2 | Resulting labelled data after Classification from sample real life Dataset. . . | 61 |
| 4.3 | Close-ups of the most prominent issues present in a sample real life Dataset faced by the classifier. | 62 |
| 4.4 | Optimisation results of a sample real life Dataset. | 64 |
| 4.4 | Optimisation results of a sample real life Dataset (cont.). | 65 |
| 4.5 | Comparison of Classification and Optimisation results of a sample real life Dataset. | 65 |
| 4.5 | Comparison of Classification and Optimisation results of a sample real life Dataset (cont.). | 66 |

LIST OF TABLES

| | | |
|-----|---|----|
| 2.1 | Data Acquisition Comparison between Laser Scanning and Computer Vision (Image-based). | 11 |
| 2.2 | Data Processing Comparison between Remote Sensing Data Sources. | 28 |
| 3.1 | Detailed information of manually labelled point cloud datasets. | 42 |
| 4.1 | Detailed information of manually labelled ground truth subsets. | 60 |
| 4.2 | Detailed information of classification metrics. | 61 |
| 4.3 | Detailed information of optimisation metrics. | 64 |

LIST OF ALGORITHMS

| | | |
|---|--|----|
| 1 | Random Forest | 49 |
| 2 | Progressive Morphological Filter | 56 |
| 3 | Euclidean Clustering | 58 |

ACRONYMS

| | |
|--------|--|
| AirSim | Aerial Informatics and Robotics Simulation |
| ALS | Airborne Laser Scanning |
| ATV | all-terrain vehicle |
| CGAL | Computational Geometry Algorithms Library |
| CHM | Canopy Height Model |
| CNN | Convolutional Neural Network |
| DEM | Digital Elevation Model |
| DSM | Digital Surface Model |
| DTM | Digital Terrain Model |
| EOL | End of Life |
| FoV | Field of View |
| GNSS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| GPU | Graphics Processing Unit |
| GSD | Ground Sample Distance |
| HT | Hough Transform |
| IMU | Inertial Measurement Unit |
| LAI | Leaf Area Index |
| LiDAR | Light Detection And Ranging |
| mAcc | mean Class Accuracy |
| MLP | Multi-Layer Perceptron |
| MRF | Markov Random Field |

ACRONYMS

| | |
|--------|--|
| nDSM | normalized Digital Surface Model |
| NDVI | Normalized Difference Vegetation Index |
| NED | North-East-Down |
| NIR | near infra-red |
| OA | Overall Accuracy |
| OFF | Object File Format |
| PCD | Point Cloud Data |
| PCL | Point Cloud Library |
| PCNN | Pulse-Coupled Neural Network |
| PDE | Partial Differential Equation |
| PLC | Power Line Corridor |
| POS | Position and Orientation System |
| RANSAC | RANdom SAmple Consensus |
| ReLU | Rectified Linear Unit |
| ROS | Robot Operating System |
| SAR | Synthetic Aperture Radar |
| SfM | Structure from Motion |
| SVM | Support Vector Machine |
| TPU | Tensor Processing Unit |
| UAS | Unmanned Aerial System |
| UAV | Unmanned Aerial Vehicle |
| VG | Voxel Grid |
| VTOL | vertical take-off and landing |

INTRODUCTION

The present chapter provides an introduction to the work that has been performed in this dissertation. The scope of this work is first explained in **section 1.1**, followed by the motivation that led to it in **section 1.2**. The most recurrent obstacles found in related work in **section 1.3**, followed by the approach to solve them presented in **section 1.4**. Next, the proposed method to achieve the goal of this work is introduced in **section 1.5**. Finally, the document organisation is briefly explained in **section 1.6**.

1.1 Scope

Electricity is a basic need for the efficient functioning of contemporary societies. To provide a reliable and continuous distribution of electricity to its customers, power distribution companies must realise regular inspections of the power line network [25, 52]. In forested countries, a huge part of the electrical network is located within the forest. Power line maintenance not only involves the monitoring of power line components, but of the surrounding objects as well, including buildings and vegetation [28].

Vegetation brings several contributions to the environment like providing habitat for wild animals, certain aesthetic benefits and tree shades can also regulate temperatures [52]. Nevertheless, it is important to manage vegetation growth, because overgrown trees can interfere with power lines, which can, consequently, result in short circuits, blackouts, or even worse, bushfires. Vegetation management aims to keep the healthy low-growing plants while removing overgrown trees that can be potentially hazardous to the power lines [21]. Falling trees, especially during natural disasters and poor weather conditions, can damage the power line components and cause power outages. Tree crown snow-load can make the trees bend and collide with power lines. Moreover, the power line components naturally deteriorate, be it due to corrosion, insulator cracks or mechanical

damage in the conductors. Therefore, it is important to not only monitor the vegetation around the electrical network, but the power line corridors (PLCs) as well [25].

In addition, certain countries, like Portugal [36], can impose strict laws related to fuel management in forest areas, and even enforce grave penalties when these laws are not respected. Fuel management bands are defined by each countries' legislation, with widths proportional to the objects surrounding vegetation (i.e. buildings, power lines) and taking into account the potential risk to cause a forest fire. In private lands this responsibility lies on the landowners. However, in public areas the responsibility lies on the entities operating the power lines. Any vegetation that can be ignited is considered as fuel and can provoke a forest fire. Vegetation crossing a fuel management band is considered as potentially dangerous and must be trimmed down, removed partially or completely. The result of this procedure is called a fuel modification area.

1.2 Problem

Conventional approaches of power line monitoring include either field or airborne surveys. Field-based monitoring can be done for longer periods of time, which, consequently, increases the problem detection rate. Moreover, it can provide crucial information about forest structures. However, this process is not only very labor-intensive and costly, but it is also dependant of subjective human visual interpretation, which makes it prone to error [2, 28, 32, 45, 48, 49]. Airborne monitoring, using, for example, a helicopter, has a lower problem detection rate, since this method is limited by the helicopter's speed and the crew's ability to detect problems at such high speeds. Furthermore, their long observation distance will, consequently, have negative impacts on the resolution of the retrieved data. In addition, this method is also dependant on a human's visual observations [8, 28].

1.3 Motivation

Many approaches have been studied in the last years aiming to refine the efficacy and reduce the workload of power line monitoring subsequently eliminating the need for power distribution companies to acquire and operate the monitoring equipment. These approaches include: camera tracking systems to acquire data, line inspection robots, and unmanned aerial monitoring. Moreover, new automated data processing algorithms have emerged, for vegetation detection, canopy height assessment and feature extraction [31]. The usage of digital cameras and laser scanners for visual inspections has been vastly studied in the last two decades, providing several benefits when compared to the traditional methods. The most important one being the fact that these approaches have distinct data collection and analysis procedures, which, consequently, reduces data collection costs substantially while improving data analysis. Nevertheless, data analysis is still fundamentally a manual process [28].

1.4 Solution

Remote sensing techniques can cover wide areas with low efforts, thus making it more efficient than traditional techniques. Data processing is increasingly more an automated process, using computer vision algorithms, making it independent of subjective human visual interpretation [25]. Optical images, Synthetic Aperture Radar (SAR) images and Airborne Laser Scanning (ALS) data are some of the approaches that have been most commonly studied in the last few years. Optical and SAR images have been recently used for PLC monitoring; however, these sensors are susceptible to occlusions. ALS has been used to overcome this issue and provide a more efficient inspection, although it also has its own disadvantages. The standard ALS platforms (i.e. helicopters and fixed-wing aircraft) can be quite costly and inflexible [8, 32].

Unmanned Aerial Vehicles (UAVs) equipped with a Global Positioning System (GPS), an Inertial Measurement Unit (IMU), laser scanners and optical cameras provide an efficient solution to traditional remote sensing platforms, with substantially lower costs. Moreover, it presents close-range data acquisition, flexibility and potential for customisation, with reduced time cost and human resources, especially for inspections in difficult terrains [8].

1.5 Proposed Method

This study will focus on the detection and classification of vegetation for power line monitoring using LiDAR-based point clouds, in other words, a collection of multi-dimensional points, acquired with a UAV platform, as well as computer vision algorithms for data processing. This procedure involves: LiDAR data acquisition, detection and classification of vegetation and electrical structures, followed by a post-processing operation which optimises the classification results.

1.6 Document Organisation

This document comprises the current introduction and four other chapters, and is structured as follows:

- **Related Work** - This chapter presents the state-of-the-art for vegetation management and power line mapping using remote sensing techniques. A brief description of these techniques is presented, including their advantages and disadvantages, followed by a comparison of their performance;
- **Method** - In this chapter the proposed remote sensing method is described. Furthermore, external resources to achieve the specified goal are also briefly explained. In particular, the proposed method consists of a UAV-based approach, equipped with a LiDAR and the support of computer vision algorithms. This method starts

with the data acquisition process, followed by the classification of objects into their respective labels, power line structures or vegetation, otherwise known as semantic segmentation. Lastly, optimisation algorithms are used to improve classification results;

- **Results** - This chapter presents the visual results attained from the classification process as well as the posterior optimisation, displaying the most important evaluation metrics. A comparison of results and their analysis is also shown.
- **Conclusions and future work** - This chapter presents the conclusions attained from the literary research and their importance during the development of this work, followed by the conclusions of the proposed methodology including its benefits and limitations and the objectives that have been accomplished. To conclude, possible future work is presented, including objectives that were not fulfilled, and other improvements to algorithms and methodologies that could be done, in order to ameliorate the results obtained in this work.

RELATED WORK

The objective of this chapter is to introduce the concepts of point cloud data acquisition and processing by presenting the most relevant related work found in literature. In the first section, **section 2.1**, data collection methods using both laser scanning and computer vision techniques will be presented, including their benefits and limitations, as well as a brief comparison of both approaches. In the following section, **section 2.2**, several methods of data analysis will be presented: Synthetic Aperture Radar (SAR) Images, Optical Satellite Images, Optical Aerial Images, Thermal Images, Mobile Laser Scanning (MLS), Airborne Laser Scanning (ALS) and Unmanned Aerial Vehicle (UAV) data. For each technique, a brief explanation of how they work will be presented, followed by their advantages and disadvantages, and, lastly, an exposition of the works found in literature using said technique and a brief summary of what these works have achieved. To conclude the section, a brief comparison of the different methods mentioned is also shown.

2.1 Data Collection

Two methodologies have been primarily used for data collection in power line monitoring and vegetation management applications: (1) laser scanning and (2) computer vision algorithms to process still camera imagery. Laser scanning is the process of computing the return time of an emitted laser pulse when it intercepts certain object which can consequently be used to calculate a range measurement and obtain a 3D point cloud. Image-based point clouds, on the other hand, are generated with computer vision/photogrammetry techniques [48]. Further explanation about how these methodologies work, as well as their benefits and limitations, is presented in the upcoming subsections.

2.1.1 Laser Scanning

Laser scanning (also known as Light Detection And Ranging, LiDAR) is an active remote sensing technique whose potential for extraction of important vegetation properties at both forest stand and individual tree level, as well as for power line mapping, has been vastly explored in the last decades [41].

Laser scanners record the backscattered echoes of short pulses of electromagnetic radiation (typically 4-10 ns), be it in the infrared or visible bands of the electromagnetic spectrum, emitted at a high frequency (typically 50-150 kHz). When a surface intercepts the laser pulse, part of its energy can be backscattered. The sensor captures the backscattered signal and the distance between the laser scanner and the intercepted surface is calculated taking into account the flight duration, as well as the velocity of the laser pulses. Moreover, the angle of deflection for each pulse is also recorded, as well as the intensity information which is quantified as the amount of backscattered energy [46]. A synchronised Position and Orientation System (POS), which is typically formed by a GPS system and an IMU, provides the position and orientation of the sensor system [46].

For each laser pulse emitted, laser scanning platforms can usually record up to four return echoes (i.e. first echo returning from vegetation or the electric network and last one from the terrain). Surface types affect the quantity of backscattered echoes as well as their height differences, which can be used to interpret data [28].

From LiDAR data, three crucial models for vegetation management can be created: (1) Digital Terrain Model (DTM), also called Digital Elevation Model (DEM) in certain studies, which provides the elevation of the ground or the Earth's surface, obtained by isolating ground points; (2) Digital Surface Model (DSM) provides the surface top, which includes the Earth's surface, as well as treetops, power lines, or other objects; (3) Canopy Height Model (CHM), also called a normalised Digital Surface Model (nDSM), is the result of the interpolation of LiDAR data when the return heights are normalised to a DTM [24]. In other words, it results from the subtraction of DSM and DTM and provides the height of the referred objects.

CHMs can, when used on the tree level, facilitate the detection and delineation of individual trees. Nevertheless, individual tree and crown extraction can still pose a challenge to CHMs. Incorrect crown shapes could be a result of CHMs crown delineation, which can overlap with other trees. Another reported issue is tree height underestimation, due to incomplete LiDAR data or tree crowns' permeability to LiDAR which can consequently generate incorrect CHMs. Thirdly, CHM will usually present surface imperfections which require correction using smoothing techniques, and will consequently have negative impacts on the detection and delineation of individual trees. Lastly, treetop locations might be moved in CHMs from their position in the raw point cloud due to rasterizing [24, 42].

PLC mapping with LiDAR data typically involves classification or segmentation of points into distinct surfaces (i.e. vegetation, electric network or terrain), followed by the modelling of wires strung between successive pylons. Since the return point number

is far lower than the input point number a lot of approaches start with pylon detection which consequently allow a quicker wire extraction since solely non-ground points have to be processed. Nevertheless, in forest areas, due to the large amount of trees and their shapes, they could exhibit similar characteristics as pylons thus making pylon detection to support wire extraction counter-productive [2].

Power line extraction methods using LiDAR can be, essentially, divided into two types: (1) supervised and (2) unsupervised [2, 32, 56]. Supervised classification uses classifiers like Support Vector Machine (SVM), Random Forest algorithms and JointBoost based on the geometrical properties of laser points [56]. It can be highly accurate for small areas, but it requires a large training dataset which is generally difficult to obtain in forest areas, since pylons are a minority compared to vegetation and even wires [2, 56]. Unsupervised classification is based on statistical analysis or Hough Transform (HT) and clustering algorithms [56]. It is usually based on the vertical and horizontal properties of power lines [56]. Since wires linking consecutive pylons have linear properties, the non-linear adjustment of catenary curves can be used to detect wire candidate points [56].

These methods can be further divided into two categories: (1) grid-based and (2) point-based. Grid-based methods interpolate 3D point cloud data (height and laser intensity) into a 2D grid space. Nevertheless, these methods will be prone to error when multiple objects overlap each other vertically [2]. Point-based methods analyse and label all 3D points individually thus overcoming the limitations of grid-based methods, but with increased computational costs. These methods can be further divided into statistical analysis, wire extraction based on line geometrical properties using HT, and RANdom SAMple Consensus (RANSAC) algorithms [2].

2.1.2 Computer Vision

With the improvements of digital aerial cameras, it is now possible to acquire multiple overlapping images in a less laborious way than before, with substantially reduced costs, from the acquisition to the processing of these images. Vegetation management can benefit from overlapping images, since it allows multi-view matching, which can overcome certain canopy occlusions while improving geometric accuracy. Furthermore, improvements in computer technology have reduced the complexity of image matching algorithms. These improvements result in the generation of image-based point clouds which can be very similar to ALS point clouds and provide more accurate and detailed DSMs. Image-based point cloud and DSM require various overlapping airborne images with high-spatial resolution, which is characterised by the Ground Sample Distance (GSD) (in other words, the minimum identifiable object size) and is affected by the flying height and sensor characteristics [53].

Stereophotogrammetry using image-based point clouds is a process similar to a human's depth perception with normal binocular vision. It follows the principle of parallax, in other words, the apparent relative position variation of a motionless object, due to

a change in the angular viewpoint. Its principle is to identify common points in each image, according to their radiometric and geometric similarities, thus creating a line of sight (or ray) between the camera viewpoint and these points. When two rays intersect, it is possible to obtain the three-dimensional position of said point using triangulation. The same process applied to all the points corresponding to the object or surface of interest can be used to generate a DSM [53].

There are two commonly used automated photogrammetric image matching methods: (1) area-based and (2) feature-based. Area-based methods typically search for a match between images using a reference window of pixels. Feature-based methods use basic geometric shape matching, like lines and polygons. Image processing software programs nowadays generally use both methods. A special case of area-based methods is Global Image Matching, which tries to find a match for each pixel in one photograph, providing more detailed and accurate results, but also very computationally intensive and slow. Semi-Global Matching is a hybrid method that has emerged, combining both area-based and Global Image Matching methods, which uses a cost function to evaluate the most efficient method for each area of the image [53].

Multispectral imaging is a remote sensing technique which uses photographs in the visible and infrared bands of the electromagnetic spectrum. However, this technique has certain limitations, such as, it cannot differentiate certain vegetation elements (i.e. branches, leaves, etc.) at different heights, and, the received signals are highly dependant of the light conditions, as well as the geometric distribution of vegetation [46]. Moreover, certain important tree parameters, such as leaf area index (LAI), may still require models to be accurately characterised. Some of these limitations can be overcome by the application of 3D imaging techniques instead, which allows the creation of certain models that can separate ground surface from different vegetation layers. The most typically used technique to retrieve 3D information is multi-image matching which finds common points in several 2D images taken from different angular viewpoints [46].

Another computer vision technique is Structure from Motion (SfM), which allows the possibility to generate high density 3D point cloud data from the reconstruction of high resolution images, obtained from different angular viewpoints, like traditional stereoscopic photogrammetry. Using image matching algorithms, it is possible to identify certain features in these image sequences. These images can be obtained using inexpensive Unmanned Aerial System (UAS) platforms, which can provide an alternative (or complement) to ALS datasets [19].

Existing studies of power line inspection and monitoring using 2D image-based methods involve information like uniform brightness, straight line characteristics, geometric shapes, among others. Several studies use HT for power line extraction from imagery. Another line detection method is to analyse the uniformity of conductors in the full extent of the power line. Methods to three-dimensionally reconstruct the wires, from the 2D imagery, include stereo vision techniques used for pylon identification, line-segment based pylon models and catenary curves, be it automated or semi-automated manners [8].

2.1.3 Comparison of Approaches

Several studies have presented the production of important forest attributes, like height and canopy dimensions, using both LiDAR-based and image-based point clouds [48]. One example being Dandois and Ellis (2010) [19] who generated tree CHMs using image-based point clouds, obtained via a DTM acquired from an existing LiDAR dataset.

Bohlin et al. (2012) [53] applied statistical analysis to an image-based DSM, with heights normalised utilising a DTM, obtained with ALS data, to estimate tree height and stem volume. The authors hypothesised that vegetation attributes obtained with point clouds derived from imagery would be less accurate than the ones obtained with ALS-based point cloud, since ALS can better penetrate the canopy. This study concludes that lowering the flying altitude does not significantly ameliorate the accuracy of canopy estimations.

Järnstedt et al. (2012) [53] make a comparison between forest parameter estimates obtained from both ALS and image-based DSM. The authors concluded that ALS achieved greater accuracies than imagery, for the estimated parameters (mean height, dominant height, basal area, volume and diameter).

Dandois and Ellis (2013) [48] generated SfM-based terrain models which were positively comparable to LiDAR-based models. In this study, the authors concluded that, in certain circumstances, image-based point clouds can capture a more detailed representation of the upper canopy than laser scanning. Nevertheless, imaging technology cannot penetrate into the lower canopy layers like laser scanning, meaning it will not be able to produce as relevant data about them and the terrain as laser scanning. Furthermore, typically, imaging technology needs an externally generated DTM (i.e. LiDAR dataset), since 3D location estimation requires visible ground terrain information from several locations.

White et al. (2013) [53] compare ALS and image-based data for vegetation monitoring using an area-based approach. This study concludes that the advantages of imagery are: **flight planning** since ALS maximum flying altitude is limited by the power of laser pulses. Imagery can achieve higher altitudes, thus increasing the field of view, which in turn increases the spatial coverage; **flying time and covered area**, since image platforms can fly higher and quicker than ALS, which, for an equal flying time, can cover a wider area; **resolution**, since, for the same given cost, imagery can obtain much greater point densities, as a function of the GSD and the number of independent 3D pixel matches. On the other hand, the advantages of ALS point cloud are: **flying conditions**, since ALS is insensitive to poor illumination conditions, it can fly for more hours per day. Poor lighting and angular viewpoints, as well as shadows and occlusion are a limitation of image acquisition, which in turn impacts the accuracy of image matching algorithms; **efficiency**, since it takes less time to acquire and process data. Nevertheless, with computer technology improvements, this gap is narrowing; Furthermore, there are certain important parameters where both have their advantages and disadvantages, which are: **outputs**,

since, from ALS data, it is possible to generate both a DSM and DTM, while image-based data is only able to generate a DSM. However, it improves visual interpretation of species and it provides other important parameters to forest inventories; **accuracies**, since both techniques have achieved comparable results for well-defined surfaces.

Jensen and Mathews (2016) [19] present a comparison between SfM-based and LiDAR-based DTMs, as well as an evaluation of the utility of the generated model for tree canopy height estimation. In this study, the authors concluded that canopy height estimates are independent of terrain model source data, with a high dispersion present in the acquired data, which confirms the potential of imagery to characterise both the ground and vegetation structure as an inexpensive replacement or complement to LiDAR data.

Wallace et al. (2016) [48] compare ALS and SfM point clouds captured with a small-UAV platform to capture the structural properties of a forest with various canopy layers, providing the dissimilarities in data acquisition, processing and collection of forest canopy structural information of each technique, as well as their advantages and disadvantages. In this study, the authors concluded that both SfM and ALS can accurately represent the upper forest canopy layer. Both technologies were able to capture the properties of the lower canopy layers. However, ALS provides a more accurate estimation of the vertical distribution of vegetation, since it can better penetrate into the lower canopy layers. On the other hand, SfM cannot penetrate dense canopy parts, which means the mid to lower canopy information is not as accurate.

Verma et al. (2016) [45] present a comparison between two remote sensing techniques (airborne LiDAR and spaceborne multispectral imagery) to estimate 3D derivatives like canopy volume. The LiDAR processing involved a CHM generation and respective segmentation into individual tree canopies using representative features. The spaceborne images were captured by the commercial satellite WorldView-2. Tree measurements were performed either manually or with the use of the software ArcGIS version 10.

Table 2.1 summarises the most important benefits and limitations of each approach. A detailed comparison between these methodologies can be found in the work of White et al. (2013) [53].

Table 2.1: Data Acquisition Comparison between Laser Scanning and Computer Vision (Image-based).

| Criterion | Advantage | Explanation | References |
|-------------------|---------------------------|--|---|
| Collection | | | |
| Spatial Coverage | Imagery | Greater field of view, and consequently spatial coverage | White et al. (2013) [53]. |
| Conditioning | Laser Scanning | Insensitive to poor illumination conditions, shadows and occlusions | White et al. (2013) [53]. |
| Analysis | | | |
| Resolution | Imagery | Greater point density for the same cost, due to smaller GSD | White et al. (2013) [53]. |
| Results | Laser Scanning Imagery | Laser Scanning can better penetrate the lower canopy layers. On the other hand, imagery provides a more complete depiction of upper canopy layers. Nevertheless, it requires an externally generated DTM (i.e. LiDAR-base) | Dandois and Ellis (2010) [10, 48]; Dandois and Ellis (2013) [48]; White et al. (2013) [53]. |
| Accuracy | Laser Scanning | More accurate canopy density metrics, due to its better penetration into the lower canopy layers | Jensen and Mathews (2016) [19]; White et al. (2013) [53]. |

2.2 Data Analysis

As stated by Matikainen et al. (2016) [28], remotely sensed data sources for power line monitoring and vegetation management include "Synthetic Aperture Radar (SAR) images, optical satellite and aerial images, thermal images, Airborne Laser Scanning (ALS) data, Mobile Laser Scanning (MLS) data and Unmanned Aerial Vehicle (UAV) data" [28]. A brief explanation about the basic functioning principles of each data source will first be presented, followed by a summary of the methodologies found in literature. Detailed information and a comparison of data sources can be found in the previously mentioned work.

2.2.1 Synthetic Aperture Radar (SAR) Images

Synthetic Aperture Radar (SAR) is an imaging radar, whose region of operation is the microwave band of the electromagnetic spectrum, used to create 2D images or 3D reconstruction of objects. This radar is usually attached to a mobile platform [28, 46].

A SAR platform creates a 2D image representation of an area by emitting successive pulses of microwave radiation and recording the backscattered signal from the illuminated target scene. SAR platforms range from compact aircraft to satellites. One of the benefits of this remote sensing technique is that microwaves penetrate clouds, thus making SAR images possible to acquire in all weather conditions, which makes it a very useful technique for disaster monitoring, for example. Moreover, SAR images are independent of flight altitude [28].

SAR image pixels can provide valuable information, such as: radar backscattering intensity, backscattered signal phase and the distance between the sensor and the target. Typically, SAR data intensity information can be taken advantage of as amplitude images since remote sensing software packages can effortlessly visualise it. SAR backscattering intensity is influenced by certain parameters which will, consequently, affect its images and are divided into system and target parameters. System parameters are: (1) wavelength of the emitted signal, (2) signal polarisation and (3) image caption geometry. Target parameters are: (1) surface irregularity and dampness, (2) vegetation composition. Information about the signal phase can be taken advantage of in SAR polarimetry and interferometry. In SAR data processing, polarimetry is the area that deals with the polarisation of emitted and received microwave radiation, which can be used in object classification. Interferometry is a technique where pixel-to-pixel phase changes are converted into terrain elevation changes from two images taken from slightly different angles by satellites distanced from tens to hundreds of meters. Elevation data can also be extracted through SAR radargrammetry, related to the stereoscopic measurement of the images, using both intensity and range values [28].

Despite SAR imagery high resolution, power line components have small features that can still be unseen in these images. Furthermore, geometrical irregularities and multiple

path scattering can difficult the identification of some complex objects, like certain pylons or other power line structures, since SAR sensors have a side-looking geometry. Features are much more visible when they are parallel to the azimuth direction of the sensor [28].

Typically, SAR image-based studies have focused on mapping power lines and pylons, using airborne SAR imagery. Carande et al. (1998) [28] and Woods et al. (2004) [28] use InSAR data for this purpose. Carande et al. (1998) present a method to identify power line poles, using X band InSAR data. Woods et al. (2004) present a method to detect pylons and transmission towers from a DSM, generated using InSAR data. Vertical obstruction detection was done using morphological filters and statistical criteria. Candidate wire points were identified based on the vertical obstructions forming a linear shape.

Yang et al. (2007) [28] and Xie et al. (2014) [28] use L band airborne data for transmission tower detection. Yang et al. (2007) utilise full polarimetric data and point-like object detection. Xie et al. (2014) evaluate how certain parameters, like “linear polarisation ratio” and “degree of linear polarisation”, can be used to detect transmission towers. The results of this study show that transmission towers were clearly visible in the SAR images.

Deng et al. (2012) [28] and Lin et al. (2012) [28] use P band data for power line detection. Deng et al. (2012) evaluated the proposed method with simulated and real P band data. The authors concluded that the orientation of the power line affects the results of the detection procedure. Lin et al. (2012) show several examinations with P band data acquired with airborne circular SAR. The results of these experiments prove the applicability of circular trajectory data to detect power lines.

Some studies have also discussed how high-resolution satellite SAR data can be used for power line mapping. Schwarz et al. (2009) [28] and Yan et al. (2012) [28] use TerraSAR-X satellite images. Schwarz et al. (2009) show how a power line pylon could be detected from a time-frequency decomposition. Yan et al. (2012) besides X band SAR images from TerraSAR, also use COSMO-SkyMed satellites to study the scattering characteristics of transmission towers and power lines. This study also concluded that the orientation of power lines affects their visibility in SAR images.

Another study about the backscattering properties of power lines is presented in Sha et al. (2014) [28]. This study concluded that scattering position changes are affected by air temperature, which can be consequently related to variations in the sag of the conductors.

As previously mentioned, remote sensing using SAR data has potential to monitor power lines due to natural disasters, like Jingnan (2010) [28] who used high-resolution SAR images to study the backscattering characteristics of a transmission tower affected by icing. With this study, the author concluded that it is possible to identify deformations caused by extreme weather conditions.

A discussion about the potential usage of satellite SAR data to monitor transmission towers and power lines after natural disasters was presented by Yan et al. (2011) [28]. In

this study, height tower information, to detect collapsed or deformed towers, was utilised.

Few studies have concentrated on the potential of satellite SAR images to detect storm damage in forest areas. Eriksson et al. (2012) [28] investigated the backscattering characteristics of felled trees, caused by storm damage. This study concluded that TerraSAR X band data backscattering increased, while ALOS PALSAR L band data decreased, both using HH polarisations. Furthermore, when a DTM is known beforehand, 3D SAR techniques can be used to estimate tree heights. Potentially, this information can also be used to detect storm damage as well, when 3D data acquired before and after the natural disaster occurrence is available [28].

2.2.2 Optical Satellite Images

Optical satellite images, captured in either the visible or near-infrared (NIR) bands of the electromagnetic spectrum, are passive imaging sensors. In other words, only the energy reflected from the surface in observation is captured, thus making these images unobtainable in poor illumination conditions [28].

The distance between neighboring pixels on the ground is called ground sample distance (GSD). It corresponds to the spatial resolution of the sensor, and it ranges from low (≥ 30 m) to very high (< 1 m) resolutions. Weather conditions (i.e. cloud cover, fog, etc.), as well as shadows and occlusion, will negatively impact optical image quality. Moreover, very-high resolution imagery can be expensive, and multiple entities can compete for the same imaging resources simultaneously [21, 28].

Multiple studies have used optical satellite images to monitor vegetation close to power line corridors. Ahmad et al. (2013) [28] present a method based on multispectral satellite stereo images to identify potentially dangerous vegetation encroachment onto transmission lines.

Moeller (2006) [28] presents a method to detect vegetation that can potentially endanger power lines using multispectral and stereoscopic satellite imagery. Firstly, tall vegetation is extracted from the Normalised Difference Vegetation Index (NVDI), followed by vegetation height calculation, using the stereo images. Lastly, vegetation that might present potential interferences are mapped.

Kobayashi et al. (2009) [21] developed a software program to identify trees that can possibly interfere with power lines, from high resolution satellite imagery. Using computer algorithms, trees that can potentially endanger power lines are identified. The software framework is described by: data loading (multispectral stereo satellite images and transmission tower); transmission and power line location estimation; calculation of the danger zone; detection of healthy vegetation inside the danger zone; selection of vegetation detection threshold; calculation of stereo matching for all pixels inside the danger zone; 3D DSM generation; detection of potentially hazardous trees inside the

danger zone.

Ahmad et al. (2011) [28] propose a method to monitor vegetation and detect potential interferences. This method starts with a merging process, of both multispectral and panchromatic images, followed by a process of vegetation detection from the NVDI. Lastly, tree height measurements were calculated from stereo image matching.

2.2.3 Optical Aerial Images

Optical aerial images acquired from both helicopters and fixed-wing aircraft, in the visible and NIR bands of the electromagnetic spectrum, have also been used in some studies for vegetation management. In addition, airborne imagery can also provide valuable information about power line components [28].

Helicopters are usually used to inspect power line components, since they can fly closer to the electric network, while fixed-wing aircraft have been mostly used for vegetation monitoring. Helicopters have certain advantages, such as: they can provide sub-centimeter GSDs, hover around an area of interest and easily map it, including routes with sharp turns (i.e. power line corridors). Fixed-wing platforms, however, can fly faster, which, consequently, means they can cover wider areas [28]. Furthermore, inexpensive hobbyist aircraft, like remotely controlled fixed wing aircraft, helicopters, balloons and kites, have seen an increase in low-altitude aerial photography usage, due to how simple it is to install consumer-grade digital cameras on these platforms. Moreover, recent computer vision algorithms, using SfM procedures, can easily generate 3D geometry from multiple overlapping images, taken from consumer-grade digital cameras [10].

Just like optical satellite images, weather conditions and poor illumination are one of the main obstacles of optical aerial imagery. In addition, the quality of the images taken can be affected by vehicle motion and vibrations. The GSD of airborne imagery can be less than 1 cm or greater than 1 m. The spatial resolution of aerial images is dependent of the sensor resolution and optic quality, as well as the flight altitude [28].

Studies about vegetation monitoring in PLCs using airborne imagery have primarily concentrated on tree extraction, segmentation and classification. Identifying their species is important since it provides information about tree growth. Trees that cross a certain height threshold could potentially endanger the PLCs. Sun et al. (2006), Yan et al. (2007) and Zhang et al. (2007) [31] use stereo imagery to estimate the distance between the conductors and the trees or the ground.

Zhang and Grün (2006) [46] propose a coarse-to-fine method and pattern matching to obtain DSMs for forested areas. The main challenge to overcome, was shadow and occlusion, which makes it harder to find matching points in the lower canopy layers. This study concluded that multi-image matching techniques can provide valuable information about the top canopy layers, but its usage is very limited for the lower canopy layers.

Li et al. (2009) [28] and Mills et al. (2010) [28, 31] present methods to automatically

extract trees in PLCs, from multispectral imagery. Both studies use a Pulse-Coupled Neural Network (PCNN), whose inputs were NIR and red band reflectance ratios, followed by a morphological segmentation process to improve the output. In addition, both studies used the same PLC as well, obtaining a good detection rate and accuracy overall. However, these accuracies are negatively affected by under-segmentation. Mills et al. (2010) discuss how accurate tree height estimation is, using stereo imagery and relative horizontal position of vegetation in relation to the wires.

Using the same data as the previously mentioned studies, Li et al. (2011, 2012) [28] presents a method for tree species classification. This method uses support vector machines (SVM) and feature comparison [28].

Dandois and Ellis (2010) [10] propose an inexpensive method for vegetation monitoring combining an inexpensive hobbyist aircraft platform (kite) and SfM. Firstly, the aerial photographs are captured by the inexpensive hobbyist aircraft platform using an off-the-shelf digital camera, then the vegetation mapping is performed using Bundler, a software program which enables 3D reconstruction using multiple photographs from different angular viewpoints, by extracting keypoints in the images. This new vegetation monitoring technique is called “Ecosynth”. The point clouds generated by Bundler are then geocorrected and a statistical outlier filter was used for outlier removal. A DTM was generated using the software TerraScan, using LiDAR data (since the generation of a DTM is one of the limitations of image-based point clouds), followed by the creation of CHMs. Lastly, using the height metrics from both Ecosynth and the LiDAR CHM, predictive models for aboveground biomass density were generated.

Given good illumination conditions and high enough resolution, it is possible to identify power line components using optical aerial images, and, in some cases, even monitor their condition. This technique does have some limitations though, like poor illumination conditions, complex background and respective brightness, weather conditions and features similar to power lines. The direction of the camera will also influence what is captured by the images. Certain vertical components can be hard to identify from vertical images, even for a human operator [28, 31]. To identify certain components then, oblique images might be needed. Furthermore, it can be challenging to model conductors in 3D because they have uniform intensity and do not exhibit any distinguishing properties in airborne imagery, which makes it harder to find matching points in different images [28].

Some studies have focused on the extraction of power line conductors using optical aerial images, following the same workflow. Firstly, candidate power line points are identified, followed by line extraction from the candidates. This process can be improved by gap filling and noise removal [28]. Ye et al. (2013) [28] use HT to obtain an initial seed line, followed by particle filtering.

Li et al. (2010) [28] use template matching to identify conductors. Each image has an expected number of conductors as input. Images with incorrect number of conductors were manually checked. Besides conductor detection, spacers along the conductors were

identified using Gabor filtering and a connected component labelling algorithm.

A few studies about pylon extraction from aerial images have also been done, focusing mostly on poles and towers. Whitworth et al. (2001) [28] present an apparatus for the surveillance of wood pole components, which consists of a steerable camera on-board a helicopter. Golightly and Jones (2003) [28] present improved algorithms for pole and cross arm smooth tracking. Jones et al. (2003) [28] used Gabor filters and Radon transform to not only detect wood poles, but also the location of the base and top of the pole and cross arm.

Sun et al. (2006) [28] used airborne imagery taken with a fixed-wing aircraft to extract wood poles and cross arms or pole tops. Candidate pole and cross arm points were identified using color and intensity information, then matched to one another to locate poles in images.

Tilawat et al. (2010) [28] used information about the high concentration of wires in tower positions to identify transmission line towers using aerial imagery. Image acquisition in both Jones et al. (2003) and Tilawat et al. (2010) was done using a helicopter.

Detection and monitoring of insulators using aerial images have also been studied, mostly focusing on the insulators of high voltage power lines. Wu et al. (2012) [28] and Wu and An (2014) [28] use active contour models to extract insulators. Oberweger et al. (2014) [28] use learning and voting procedures, as well as circular descriptors, to detect cap and pin insulators. Liao and An (2015) [28] use feature matching algorithms to detect insulators.

2.2.4 Thermal Images

Thermal images detect infrared radiation from an object, based on its temperature, which can be described by the Stefan-Boltzmann law. Typically, thermal imaging spectral range goes from 3-5 μm (mid-wave infrared) to 8-14 μm (long-wave infrared), which is the most commonly used atmospheric thermal window of thermal cameras. A special case of thermal cameras are cooled cameras, which have better sensitivity and image quality, but are also much more expensive than uncooled cameras. Airborne thermal images spatial resolution can go up to 5 cm [28].

Not many studies have focused on power line monitoring using thermal imaging, being mostly focused on the assessment of electric faults in the power line. Blazquez (1994) [28] was able to detect faulty components due to unusual temperatures on high-voltage transmission line using thermal imagery. Frate et al. (2000) [28] present a method whose objective is to obtain the temperature of an overhead line and joint using thermal imagery. Qin et al. (2000) [28] discuss the heat conduction of electric power lines.

Stockton and Tache (2006) [28] show that it is possible to detect relevant electrical

faults in power lines. However, it is not possible to obtain accurate temperature readings, even from short distances.

To conclude, it is possible to detect electric faults using airborne thermal images. However, it is not possible to obtain accurate measurements of the temperature of the components. Nevertheless, thermal imaging is generally used before an inspection, to prevent expensive service interruptions [28].

2.2.5 Mobile Laser Scanning (MLS)

Mobile Laser Scanning (MLS) is an active remote sensing technique, where a mobile mapping system, constituted by positioning, navigation and imaging acquisition sensors, is deployed on a land-based mobile platform, like a car, all-terrain vehicles (ATVs) or even a personal laser scanning system, which is basically a LiDAR deployed on a specialised backpack [25, 28].

MLS data can be more or less accurate depending on certain aspects, such as vehicle speed, platform configuration (i.e. number of sensors), the relative position between the vehicle and the object in observation, the capability to record multiple echoes and the season of the year when the data is acquired (tree leaf numbers are dependant of season) [41].

MLS with dense point clouds is pertinent for PLC, and it can complement the conventional ALS, or the contemporary UAV-based laser scanning. MLS can provide accurate data about the PLC since it operates on the terrain, in close proximity to the electric network, especially when it is covered by the forest canopy, and it also has high point density. Moreover, it has a small footprint, which provides a detailed mapping of pylons and other electric components [25].

MLS coverage has been mostly studied in urban areas where it is more difficult and inefficient to fly with a UAV or a helicopter. However, data gaps appear everywhere, especially in heavy forests. These areas can present some limitations to MLS however, like terrain irregularities causing mobility restrictions and Global Navigation Satellite System (GNSS) visibility, which makes the power line component extraction more challenging outside the road network, reason why it has not been extensively studied yet [25].

Studies focused on vegetation management include Liang et al. (2014) [28] who present a tree detection method, Puttonen et al. (2011) [28] who propose a tree species classification, both using MLS data. An approach to extract urban trees using a combination of MLS and images was studied in Yang et al. (2012, 2015) [28].

Rutzinger et al. (2010) [41] presents an automated workflow to detect and reconstruct individual trees using MLS point clouds, which can be summarised as a reduction of the amount of data while keeping important shape information for tree modelling. Firstly, using a connected components algorithm, individual trees are extracted. Then, using a 3D alpha shape algorithm, it is possible to reduce the point cloud data substantially,

by the simplification of tree shapes. From the alpha shape point clouds, the minimum required tree model parameters can be extracted, like height, crown width and shape, as well as, stem width and height. The generated models are, lastly, compared to tree photographs and the original point cloud data.

Several studies have proposed power line component mapping using MLS data, however, most of them have been done on urban or road environments, including Lam et al. (2010) [28] who propose a method to extract pylons using RANSAC, followed by power line modelling, by extracting all points inside a box between two consecutive pylons.

An approach to monitor power lines in urban and suburban areas using digital cameras deployed on a kinematic land-based vehicle was presented by Murthy et al. (2011) [28]. Pole tracking was done using template matching, while broken insulators were classified using a hidden Markov model.

Kim and Medioni (2011) [25, 28] proposed a method to extract power lines and poles using both ALS and MLS data. Firstly, ground and building points are removed, then linear points in the 3D space were extracted using tensor voting. Using a surface growing algorithm, the wires and poles were extracted from the points with linear properties. Linear points with their main component close to horizontal were used for power line growing, while points close to vertical were used for pole growing.

A power line extraction and modelling method was proposed by Guan et al. (2016) [25, 28] and Cheng et al. (2014) [25, 28]. Using multiple filters, HT, segmentation and power line clustering for power line extraction. For power line modelling, Guan et al. (2016) used catenary curve fitting, while Cheng et al. (2014) used local line and parabola fitting.

Yadav and Chousalkar (2017) [25] propose a method for power line classification and reconstruction using MLS data. Candidate power line point classification is achieved according to the point's vertical distance to the terrain surface, utilising 2D gridding. Vegetation and building points are removed based on 2D point density. Horizontal lines are extracted by the application of an HT. Lastly, single power lines are segmented, followed by a second order curve fitting, for gap filling and 3D reconstruction.

Lehtomäki et al. (2019) [25] propose a method to automatically map the power lines outside urban areas. Firstly, the feasibility of MLS outside the road environment is demonstrated, then a novel algorithm is presented to automatically extract PLs from MLS data. The proposed algorithm starts with power line horizontal location extraction. Power line points are extracted as connected line segments. Pole extraction and power line candidate point classification follow the same principles as Kim and Medioni (2011), followed by a voting-based approach to find horizontal lines from the candidate points, similar to Cheng et. al (2014), Guan et. al (2016) and Yadav and Chousalkar (2017). Candidate point classification linearity measure follows the same method as Lehtomäki et al. (2016) [25] and horizontal lines are identified using RANSAC, another voting method to detect shapes from point cloud data, followed by a segmentation method to obtain the connected power line segments. To remove false detections due to large gaps in the

point cloud data, as well as to retrieve the power lines, a novel method was used, named connected power line search.

2.2.6 Airborne Laser Scanning (ALS)

ALS is an active remote sensing technique constituted by an airborne platform and a LiDAR system [28]. ALS platforms are defined by a small-footprint (< 1 m) and the quantity of recorded return pulses (1 to 5 per laser pulse) and can achieve measurements with sub-meter accuracy. For an area-based approach, a pulse density of "0.5-1 pulse per square meter" [53] is usually sufficiently accurate. Dense forested areas or areas with steep terrain, however, require greater pulse densities [53].

In addition, the point density in ALS is usually about tens of points per square meter, which can go up to hundreds of points per square meter, using helicopters. ALS data has an absolute planimetric accuracy of, approximately five to ten centimeters, while its absolute height accuracy is, approximately, two to five centimeters, thus making it a viable option for detailed mapping and monitoring of power lines, as well as to reconstruct 3D conductors and vegetation. Like optical aerial images, helicopters are usually used to inspect power line components [28].

The most common approaches to vegetation monitoring using ALS point cloud data are either dependant on single tree delineation and classification or area-based measurements of canopy metrics.

In the area-based approach, tree-level measures inside the region of interest are first extracted from ground plots, then predictive models are created. From the clipped normalised ALS point cloud, metrics are calculated. Then, the same metrics are calculated for the entirety of the acquired ALS data, using the previously created predictive models [53]. An example of this approach is presented in Andersen et al. (2003) [50] who use a group of regression functions to estimate vertical canopy structure.

Leeuwen et al. (2010) [24] proposed a method to generate a tree-level canopy model, where forest canopy is described as a series of cones fitted to the raw data, correcting the negative height bias, named the Parametric Height Model (PHM), which can delineate tree crowns with geometric operations. The proposed method starts by separating ground and non-ground points, followed by the generation of a gridded DTM. A CHM is then obtained from the normalised non-ground point cloud, following Ferster et al. (2009) [24]. Using several height thresholds, of small variance, a binary image stack was generated from the CHM, to locate the local maxima. Fitting cone-shaped objects at the local maxima positions, using a modified HT, it was possible to detect individual shapes. Lastly, the PHM was generated, from the optimisation of the fit.

Studies focused on individual tree detection and classification include Pyysalo and Hyypä (2002) [50, 51] who present a tree crown reconstruction method, based on the

location and crown size of individual trees, by first extracting the raw points of the tree, and, consequently, obtaining the tree height, as well as, the average radius and height of the crown at dissimilar elevations.

Mosdorf et al. (2004) [26, 41, 54] present a method where individual tree detection is performed using the k-means clustering algorithm. However, the seed points of the presented clustering method are extracted from the LiDAR-based CHM, which means this method is not entirely relying on point cloud data.

Popescu and Wynne (2004) [26] and Tiede et al. (2005) [26] propose local maximum filtering techniques to detect and extract the most important parameters of individual trees. Following the local maximum algorithm, Tiede et al. (2005) applied a region growing algorithm to delineate tree crowns.

Chen et al. (2006) [26] propose a method to separate individual trees using "marker-controlled watershed segmentation" [26], using the detected tree tops as markers to improve accuracy. Koch et al. (2006) [26] proposed a "combination of a pouring algorithm, knowledge-based assumptions of the shape of trees and crown-edge detection" [26] to delineate individual tree crowns.

Wang et al. (2007) [51] present the first procedure to three-dimensionally reconstruct individual trees, using the LiDAR raw point cloud. Reitberger et al. (2009) [54] present an approach where the tree area is first subdivided into a voxel space, using a normalised-cut segmentation approach, then individual trees are extracted from identified graphs.

Lee et al. (2010) [26, 54] and Tittmann et al. (2011) [54] propose a method for individual tree crown segmentation. Lee et al. (2010) using an "adaptive clustering method" [26, 54], while Tittmann et al. (2011) using a "RANSAC-based approach with geometric model fitting" [54].

Li et al. (2012) [26, 54] present a "spacing-based tree segmentation algorithm" [26, 54]. Firstly, the ground and non-ground points were separated and the DTM generated. Then, the individual tree segmentation is performed, by a "top-to-bottom region growing approach, based on a spacing threshold, a minimum spacing rule and a horizontal profile of the tree shape" [26, 54]. Under and over-segmentations can be reduced by adjusting the spacing threshold.

Ko et al. (2012) [28] propose a method to identify trees endangering the power line using high-density laser scanner data. Firstly, a tree species classification process was performed, using a random forest classifier with geometric features as input data. This classification process provides valuable information about tree growth. In other words, it is possible to detect trees that surpass the minimum clearance distance to the power lines.

Zhang et al. (2015) [54] propose novel algorithms for individual tree detection and respective metric estimation, using LiDAR point clouds for urban forest inventory management. This study presents a detailed workflow for individual tree detection. Firstly, a filtering process is done, to separate non-ground and ground points, which are then used to create the DTM. Another filtering process is done, this time to the above-ground

points, to separate non-vegetation and vegetation points, using the NDVI, obtained from the hyperspectral data. Laser points touching the tree canopy surface are then separated from the points that penetrate the canopy, in order to extract individual trees. To identify the individual trees' treetop, a treetop detection algorithm is used (tree climbing method), followed by a tree boundary extraction algorithm ("donut expanding and sliding method" [54]), to delineate individual trees. All the parameters extracted from individual trees are then stored in a tree inventory database.

There are still a few studies which follow both area-based and individual tree approaches, like Wang et al. (2008) [41, 50, 51], using the LiDAR raw point clouds, present a method for both individual tree detection and delineation, as well as vertical canopy structure inspection. The vertical canopy structure provides valuable information, like the number of major canopy layers, as well as their height range. The individual tree modelling process is able to detect individual trees in the lower canopy layers, delineating the shapes and generating 3D models of the tree crowns. Firstly, a DTM is generated, by normalising the raw point heights. A height distribution probability function is created, using statistical analysis, to detect canopy layers. By extracting the 2D horizontal projection images, the detection of individual tree crowns at different heights is possible, as well as providing the voxel space of each layer. A morphological process of opening and closing is used to obtain potential tree crown contours. This process is improved by a simulation of pouring. Lastly, the 3D models of the tree crowns are obtained by constructing the 3D prisms in each voxel space.

Tang et al. (2012) [42] present a region-based level set segmentation method to implement the 3D surface reconstruction of forest canopies. Using multiple contours at different tree canopy heights, it is possible to obtain important tree parameters, as well as, to identify tree species, using the 3D reconstruction from LiDAR data, comparing it with geometric models. Moreover, individual trees in a large contour can also be extracted using the proposed method.

Several studies have also focused on power line monitoring using ALS data. Typically, power line classification studies using ALS have focused on the development of automated classification and reconstruction methods. The most common procedure is to first generate a DTM, followed by power line and vegetation point classification, then model the individual conductors in 3D. Power line and vegetation point classification is usually done with line detection or feature classification. The most commonly used features are return pulse difference, height and intensity [28].

Axelsson (1999) [28, 56] and Melzer and Briese (2004) [28, 32, 56] propose power line classification methods using HT. Axelsson (1999) utilised ALS data, based on parallel and linear 2D structure matching, using the HT and 2D line equations for line extraction, based on information about multiple echoes and intensity. Melzer and Briese (2004) propose a method based on 2D HT and 3D catenary curve fitting to extract and model

power lines from ALS data.

Clode and Rottensteiner (2005) [28] and Rottensteiner (2006) [56] propose tree and power line detection methods using the first and last pulse return differences of the laser scanner. Clode and Rottensteiner (2005) use intensity and height contrast between first and last return pulses to separate power lines and trees, based on the assumption that the elevation contrast between first and last return pulses is much greater for trees than power lines. This classification method followed the Dempster-Shafer theory.

McLaughlin (2006) [28, 32] presents a method for electric network, vegetation and other surfaces (like terrain and buildings) classification, using a Gaussian mixture model. The point cloud was previously structured in ellipsoidal neighborhoods, using the covariance matrix.

Liu et al. (2009) [2, 28, 56] use both the pulse return differences of the laser scanner and HT. The authors propose a method based on the intensity of the laser return and an improved HT to detect power lines in 2D grayscale images.

Recently, some machine learning algorithms, as well as a greater number of features, have been used for the same purpose. Sohn et al. (2012) [2, 32, 56] propose a power line and building classification method using a Markov Random Field (MRF) classifier to outline the spatial context of certain features. Guo et al. (2015) [28, 32] discuss a classification method using a JointBoost classifier and several features, followed by graph-cut segmentation to improve results.

Jwa et al. (2009) [32, 56] and Cheng et al. (2014) [56] propose methods voxel-based approaches to monitor power lines. Jwa et al. (2009) introduce a power line reconstruction method using a voxel-based piecewise line detector. Cheng et al. (2014) present an approach to detect and group power lines together, which starts by the extraction of wire points using a voxel-based method, followed by a clustering method and 3D power line fitting.

Liang et al. (2011) [56], Kim and Sohn (2011) [56] and Wang et al. (2017) [32] follow RANSAC-based approaches for power line classification. Liang et al. (2011) use RANSAC to obtain the points belonging to the line. Kim and Sohn (2011) propose a combination of RANSAC and feature extraction methods for detection of power lines, as well as a random forest algorithm for classification. Wang et al. (2017) reconstruct PLC direction using the HT with RANSAC algorithms and do power line classification using the SVM classifier using a slanted cylindrical neighborhood.

Kim and Sohn (2013) [2, 28] present a conductor, pylon, vegetation and buildings classification method. Firstly, power lines are extracted using HT, then classification is done, using random forest classifiers and spatial distribution features of laser points, obtained with spherical and cylindrical neighborhoods.

Zhu and Hyypä (2014) [56] proposes a power line classification method in forest areas, using statistical analysis. All potential power line point candidates are first selected from the dense point cloud using statistical criteria (i.e. elevation criteria, density

distribution and histogram thresholds), followed by an outlier removal procedure. These candidate points are then conveyed to a binary image, where image processing algorithms are used to remove noise and extract features. The 3D power lines are extracted by transferring the 2D power line image to a binary image as well.

Wang et al. (2018) [49] present an unsupervised wire classification method for both forest and urban areas. This method derives only linear and angular features from the PLC direction and a local neighborhood to extract power lines. The first step is a rough candidate wire point screening, by separating ground and non-ground points, followed by a PLC direction detection obtained from a layered HT, connectivity evaluation and Douglas-Peucker simplification algorithm, to remove the points that certainly do not correspond to power lines. Then, wire context-based classification, by first constructing the slant cylindrical local neighborhood of the screened point's, followed by linear and angular feature extraction, to determine if the points belong to the power lines.

Power line conductors between consecutive pylons have a catenary curve shape, so, it is possible to detect power line points by the application of a catenary curve fitting process [28]. Guo et al. (2016) [28] take advantage of consecutive pylons' conductor properties to improve power line reconstruction, based on the assumption that a group of conductors is closely equidistant, with the same direction and sag.

Jwa and Sohn (2012) [28] present a piecewise catenary curve growing method. Firstly, candidate power line points are separated using linear feature extraction. These points are then used as seed points of the growing models.

Awrangjeb and Islam (2017) [2] started by separating non-ground points, then generating a power line mask to detect successive pylons. Pylon masks were also generated, where pylons were not connected by wires, then, using a connected component algorithm the candidate pylons were extracted. Trees can then be removed, by comparison with area, shape and symmetry characteristics of pylons. Lastly, wires are extracted from the power line masks, represented as long lines within it. Using parallelism property between these lines and lines connecting any couple of candidate pylons, it is possible to remove trees with similar characteristics as pylons.

Awrangjeb (2019) [2] proposes a method to extract PLCs, pylons and wires, respectively. Firstly, from the input data, PLCs are extracted as a set of rectangular regions connected to each other. Then, pylons are extracted from this new set of points. Lastly, wires can be extracted, by separating the non-ground points between successive pylons.

Munir et al. (2019) [32] present a method to extract individual pylons, vegetation and power lines from LiDAR point cloud data using both supervised and unsupervised methods. The supervised method starts by separating ground and non-ground points. The non-ground points can be further divided into vertical (i.e. pylons and trees) and non-vertical (i.e. wires) points using the vertical profile features through the binary SVM classifier. Using shape and area characteristics, independent pylons and trees are extracted. Using the location of consecutive pylons, the span points between them are

extracted, which can be further divided into independent power line points, using the alignment properties of wires in a 3D voxel grid.

2.2.7 Unmanned Aerial Vehicle (UAV)

In recent years power line monitoring studies using UAVs have increased, since it can fly closer to the power lines and is an inexpensive alternative to helicopters. Small sized UAVs (less than 5 kg) provide an inexpensive alternative to airborne and spaceborne systems, which can, when supplied by sensors, provide cost-effective data in restricted areas [48].

In addition, technological improvements of UAV platform components and battery technologies have increased its reliability for the specified purpose. Typically, maximum flying time for a UAV ranges between half an hour and two hours, however, this time is directly related to the platform characteristics and the payload weight [28].

UAV platforms used in power line monitoring studies can be categorised into two types: fixed-wing UAVs, which are more appropriate for vegetation management due to their ability to fly faster and higher; helicopters and multi-rotor UAVs, which are more appropriate for power line surveys since they can hover closer to objects [28].

The approaches used in power line monitoring studies are similar to the previously mentioned approaches using ALS. One example is Toth and Gilpin-Jackson (2010) [28] who present a study about UAV applications for transmission line monitoring. These applications include structural inspection, vegetation monitoring and damage assessments due to natural disasters or extreme weather conditions.

Most UAV data studies, however, have focused on line detection using optical aerial images. ALS on-board a UAV platform has not been thoroughly studied yet. Katrašnik et al. (2010) [28] present a study about the application of mobile robots, where UAVs were included as flying robots, for power line inspections. Montambault et al. (2010) [28] present a study about vertical take-off and landing (VTOL) UAVs for power line inspection. However, these studies focused primarily on UAV technical characteristics, instead of specific methods.

Nevertheless, some studies present the methodologies used, such as Li et al. (2010) [28] who propose a method to extract power lines using fixed-wing UAV images. Firstly, a noise removal filter is applied (pulse coupled neural filter), followed by an HT to extract straight lines. Lastly, a segmentation process is done using knowledge-based line clustering.

Cai and Walker (2010) [28] propose a method to estimate tree heights using a new stereo image matching algorithm, applied to images acquired using UAVs and small airplanes. Castelluci et al. (2013) [28] present a method based on computer vision algorithms for pole and respective cross arm detection, using oblique colored images acquired

with a UAV.

Laurrari et al. (2013) [28] present a novel system named “RELIFO” to autonomously inspect power lines in real time, using UAV data. Using computer vision algorithms, the system is able to detect power lines and calculate their distance to vegetation or buildings.

Sharma et al. (2014) [28] present an approach to extract power lines using fixed-wing mini-UAV images. Firstly, power line points are separated, for different illumination conditions, using adaptive thresholding. Candidate power line points are obtained using a morphological operator. Lastly, power line detection is achieved using a heuristics-based approach.

Martinez et al. (2014) [28] propose a real-time method for autonomous detection of electric towers. Neural network classification was used to identify transmission towers and real-time tracking was achieved using a hierarchical tracking method.

Józków et al. (2015) [28] propose a method for 3D power line modelling using UAV images. Firstly, a point cloud is produced with dense image matching, then the conductors are modelled with catenary curve fitting. Limitations to this method include the fact that the UAV images need a high resolution and overlap, for appropriate image matching.

Studies using ALS on-board UAV include Ax et al. (2013) [28] who propose a method for vegetation management at power lines using a laser scanner deployed on a UAV helicopter. Firstly, a 3D point cloud is generated, then imported to a model of the transmission line. This model is then used to compute the distance between the trees and the conductors.

Chen et al. (2018) [8] propose a method to automatically detect clearance anomalies, based on LiDAR point cloud data acquired with a large-scale UAV platform. This method presents the described workflow. Firstly, an extraction step, which starts by terrain filtering or in other words, to separate terrain and non-terrain points using a two-step adaptive terrain filter. From the non-ground point clouds, pylons are first detected and located using a feature map approach, then point clouds are separated into spans. This step ends with the extraction of wires, according to their geometrical distribution, followed by their segmentation into clusters using conditional Euclidean clustering with linear feature constraints. The second step, modelling, is achieved by the application of "catenary curve fitting to the power line point clouds segments using two geometric shapes in two 2D dimensions, a horizontal line and a vertical catenary curve" [8]. The last step is to calculate clearance anomalies, by combining the previously constructed 3D power line catenary, a DTM generated from the terrain point clouds, and points which do not represent any power-facility objects, from the non-terrain point clouds. Trees within the danger zone are considered clearance anomalies.

2.2.8 Comparison of Approaches

LiDAR has been the leading remote sensing technique for vegetation management, using 3D information on forest vertical structure, as well as, power line monitoring. However, recently there's been an increase in the usage of passive remote sensing techniques to produce analogous information or to complement laser scanning data. This increase can be explained by the needs to control cost, current regulatory requirements or to complement the traditional vegetation inventory techniques. Likewise, some information can be hard to acquire without imagery, like individual tree species composition, maturity and health status [53]. Nevertheless, active remote sensing techniques can penetrate the vegetation more easily, which makes them more suitable for vertical vegetation structure [46].

Further explanations about the most important benefits and limitations of each approach are summarised in **Table 2.2**. In addition, a detailed comparison between these techniques can be seen in the work of Matikainen et al. (2016) [28].

Table 2.2: Data Processing Comparison between Remote Sensing Data Sources.

| Remote Sensing Data Source | Applications | Benefits | Limitations | References |
|--|---|---|--|--|
| Synthetic Aperture Radar Images | <ul style="list-style-type: none"> Disaster monitoring ; Power line and structure mapping. | <ul style="list-style-type: none"> Images independent of weather and illumination conditions; Microwaves can penetrate the canopy (to a certain extent); High spatial coverage. | <ul style="list-style-type: none"> Small features might not be visible (i.e. electric components); Difficult interpretation of complex objects (i.e. certain pylons); Topography can negatively impact, or even eliminate, vegetation backscattering; Very high-resolution imagery can be expensive. | <ul style="list-style-type: none"> Matikainen et al. (2016) [28]. |
| Optical Satellite Images | <ul style="list-style-type: none"> Vegetation management. | <ul style="list-style-type: none"> High spatial coverage; Provides multispectral data. | <ul style="list-style-type: none"> Images unobtainable in poor illumination conditions; Poor weather conditions, shadows and occlusions negatively impact image quality; Very high-resolution imagery can be expensive. | <ul style="list-style-type: none"> Kobayashi et al. (2009) [21]; Matikainen et al. (2016) [28]. |
| Optical Aerial Images | <ul style="list-style-type: none"> Vegetation management (fixed-wing aircraft); Power line and structure mapping (helicopters). | <ul style="list-style-type: none"> High spatial resolution; Ease of digital camera deployment in inexpensive airborne platforms; Flight route planning flexibility. | <ul style="list-style-type: none"> Poor illumination and weather conditions negatively impact image quality; Airborne platform velocity can affect image quality; Unable to penetrate into lower canopy layers; Difficult to create a 3D reconstruction of electric components from airborne images. | <ul style="list-style-type: none"> Wagner et al. (2008) [46]; Mills et al. (2010) [31]; Dandois and Ellis (2010) [10]; Matikainen et al. (2016) [28]; |
| Thermal Images | <ul style="list-style-type: none"> Damage assessment in electric components. | <ul style="list-style-type: none"> Potential to detect electric faults using airborne thermal images. | <ul style="list-style-type: none"> Difficult to accurately measure the temperature of the components | <ul style="list-style-type: none"> Matikainen et al. (2016) [28]. |
| Mobile Laser Scanning | <ul style="list-style-type: none"> Power line and structure mapping. | <ul style="list-style-type: none"> Detailed mapping of small areas; Accurate information about PLC, since it operates closer to the power line; Small footprint which can map electric components and pylons; Potential to fill gaps in forest inventories, left by ALS; | <ul style="list-style-type: none"> Route planning and navigation in irregular terrains; Slow process, particularly in dense, deciduous forests; Expensive usage in wide area coverage. | <ul style="list-style-type: none"> Rutzinger et al. (2010) [41]; Matikainen et al. (2016) [28]; Lehtomaki et al. (2019) [25]. |
| Airborne Laser Scanning | <ul style="list-style-type: none"> Vegetation management; Power line and structure mapping. | <ul style="list-style-type: none"> Independent of illumination conditions; Flight route planning flexibility; Provides detailed information about the forest stand, since it can penetrate lower canopy layers. | <ul style="list-style-type: none"> Requires high point density to map electric components; Inaccurate measurements of vertical poles when the airborne platform is directly above them. | <ul style="list-style-type: none"> Wang et al. (2008a) [50]; Wang et al. (2008b) [51]; Leeuwen et al. (2010) [24]; Rutzinger et al. (2010) [41]; Li et al. (2012) [26]; Tang et al. (2013) [42]; White et al. (2013) [53]; Zhu and Hyypä (2014) [56]; Zhang et al. (2015) [54]; Matikainen et al. (2016) [28]; Wang et al. (2018) [49]; Awrangzeb (2019) [2]; Munir et al. (2019) [32]. |
| Unmanned Aerial Vehicle | <ul style="list-style-type: none"> Vegetation management; Power line and structure mapping. | <ul style="list-style-type: none"> Flight route planning flexibility; Inexpensive alternative to traditional airborne platforms; Increasing reliability with technological improvements to its components and batteries; Customisation potential for multiple applications. | <ul style="list-style-type: none"> Maximum flying time still restricted by battery technologies; Emerging technology, which presents certain operation and regulatory implications | <ul style="list-style-type: none"> Wallace et al. (2016) [48]; Matikainen et al. (2016) [28]; Chen et al. (2018) [8]. |

METHOD

This chapter first presents the proposed method and the most important concepts employed, including the external resources utilised during this work. In the first section, **section 3.1**, a brief summary of the most commonly used methodologies and promising algorithms from the aforementioned work found in literature will be shown. In the contiguous section, **section 3.2**, the proposed workflow will be presented, describing each step in depth. To better illustrate this workflow, its diagrammatic representation will be displayed. In the next section, **section 3.3**, the external resources and software frameworks used during the development of this work will be briefly explained.

Further details about the proposed method, including its implementation, are then presented. In **section 3.4**, the data acquisition process to obtain simulated data will be thoroughly explained, as well as the reason behind the acquisition of simulated data instead of data acquired in a real-life environment. Next, a brief explanation of the process to mold the acquired simulated data will be presented, in order to be used as real-life data would. To better illustrate the whole process, a sequence diagram is shown. However, even though real-life data was not possible to be physically acquired, data from an open repository was also utilised and will be thus described. In the subsequent section, **section 3.5**, the data classification process is presented, starting with a brief description of the input data structures, followed by the explanation of which features are selected and how they are computed. Next, the classifier and classification functions are exposed. Lastly, the evaluation metrics utilised to appraise the performance of the classification model are presented. The last section, **section 3.6**, provides an explanation of classification post-processing including the commonly used optimisation methods, how they work and what their goal is, followed by the methods employed in this work.

3.1 Concept Overview

The objective of this study, as briefly described in the first chapter, is to propose a vegetation detection and power line classification method using LiDAR-based point clouds, acquired with a UAV platform, as well as computer vision algorithms for data processing. To date, most UAV data studies have focused on line detection using optical aerial images. Nevertheless, airborne imagery has certain limitations as previously stated, like poor weather and illumination conditions, shadows and occlusion. ALS, due to the fact that it is insensitive to these limitations, provides a reliable alternative/complement. Nonetheless, ALS on-board a UAV platform has not been thoroughly studied yet. However, works found in literature validate its applicability for the specified purpose. Essentially, two study areas are comprised in the proposed method: (1) vegetation management and (2) power line mapping.

Vegetation management, in the context of this work, starts with the detection of vegetation growing close to the electric network, followed by a tree segmentation process, to define the areas of interest. However, this process can also provide valuable information about tree height and species. Then, the horizontal distance between the trees and the power line is calculated based on the along-track position (parallel to the power line) and cross-track distance (perpendicular to the power line). Along-track position considers the line sway in irregular weather conditions. Cross-track distance is the horizontal distance between a tree's position and the power line span. If the tree is directly beneath the line, this distance is null. Lastly, the tree heights are estimated and compared to the along-track fuel management band, to conclude whether the tree growth can potentially endanger the electric network. Furthermore, identifying the tree species can also provide valuable information about tree growth [31].

Power line mapping typically involves two steps: detection of power lines, followed by power line reconstruction, where candidate power line points are extracted and modelled in 3D. Detection is typically done by point cloud structure analysis, feature extraction or straight line analysis. Reconstruction methods typically start by separating candidate power line points, followed by geometric modelling of the power line. The objective is to define and refine local models, then to obtain the complete catenary curve models [25].

- **Vegetation Detection** - Typical methods to detect vegetation using laser scanning include the generation of a DTM, by separating ground and non-ground points, followed by the generation of a CHM, from the interpolation of LiDAR data when the return heights are normalised to the DTM.

On the other hand, methods related to image-based point clouds include software to perform automated photogrammetric image matching, be it area-based (using a reference window of pixels) and feature-based (using geometric shape matching). SfM is another method which can generate high density 3D point cloud data from the reconstruction of high resolution images;

- **Tree Segmentation** - Extracting individual trees can be achieved using several algorithms, including: k-means clustering algorithm; local maximum filtering; region growing algorithms; voxel-based algorithms; RANSAC-based approaches.

Tree extraction methods from airborne imagery include: PCNN, from multispectral imagery, whose inputs are the NIR and red band reflectance ratios; SVM and spectral feature analysis;

- **Power Line and Structure Detection** - Generally, power line extraction from laser scanning data is performed through supervised or unsupervised classifiers. SVM, random forest algorithms and JointBoost are the most commonly used supervised classifiers. On the other hand, statistical analysis or HT and clustering algorithms like a voxel-based line detector are the most used unsupervised classifiers. Another promising method is to use the first and last pulse return differences of the laser scanner.

Power line and structure detection using imagery has a few limitations, such as, certain vertical components can be hard to identify and reconstruct from vertical images, even for a human operator, requiring oblique images. There are, however, certain methods used for this purpose, following a similar workflow to laser scanning methods, including: feature matching algorithms; Gabor filtering; neural network classification; connected component labelling algorithms;

- **Power Line Reconstruction** - Commonly used power line reconstruction methods found in literature are: voxel-based and RANSAC-based; using a MRF; catenary curve and linear fitting models; region growing algorithms; Douglas-Peucker line simplification algorithm;
- **Fuel Management Band Delineation** - Following the reconstruction of the electrical network and the vegetation surrounding it, the generated models can be imported to an image processing software and used to delineate the fuel management bands according to distances established in legislation;
- **Potentially Dangerous Vegetation Identification** - Succeeding the delineation of the fuel management bands, it is possible to identify vegetation crossing them, which can be potentially hazardous to the power lines.

3.2 Proposed Workflow

Based on the concepts described before, the proposed workflow for this work was established. A diagrammatic representation of this workflow is presented, in **Figure 3.1**, followed by a detailed step-by-step explanation of said workflow.

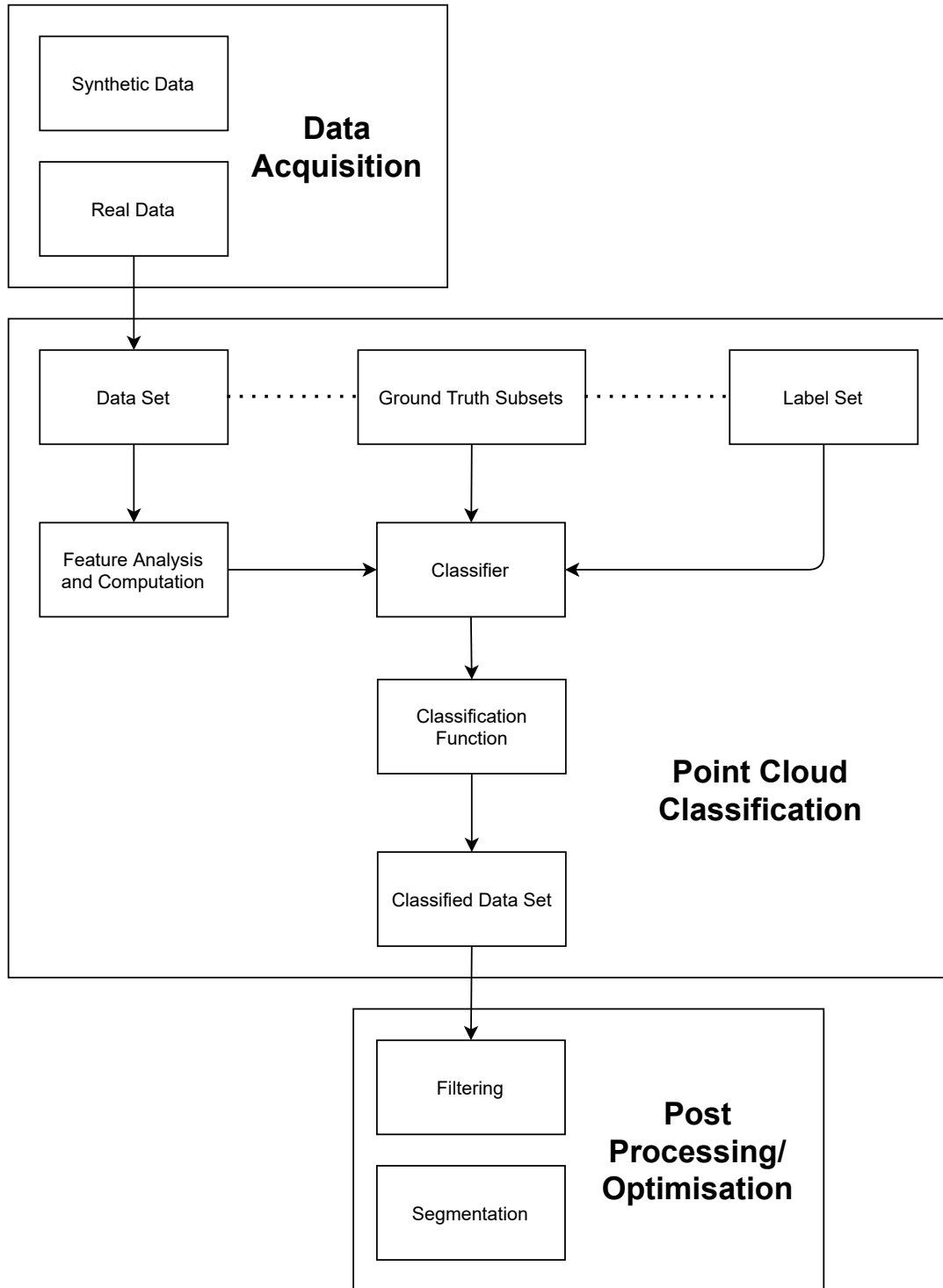


Figure 3.1: Diagrammatic representation of the Proposed Workflow.

The step-by-step explanation, of the previously displayed diagrammatic representation of the proposed workflow, is described as follows:

- **Data Acquisition:** First and foremost it is necessary to gather point cloud data to work with, whether it is captured personally or collected from an open repository. Furthermore, this data can be real or synthetic. This work will emphasise on both, providing a (relatively) simple method to produce synthetic point cloud data and presenting a promising open data repository which provides point cloud data captured in real-life environments. Further information related to the data acquisition process is provided in section 3.4;
- **Point Cloud Classification:** In this module the point cloud classification steps are contained, elaborately discussed in section 3.5. This process starts by defining the input data structures, followed by the analysis of certain predefined features using said data structures. These features are then fed to a classifier that assigns objects a class label based on the values of these features. Lastly, a classification function can be utilised to regularise the classification results. A more detailed description is given in the following bulleted list:
 - **Input Data Structures:** The input data structures that are utilised in the proposed classification method are: the previously acquired point cloud Dataset, Ground Truth subsets obtained from labelling a portion of the input Dataset and a Label Set which contains the respective labels of the objects in observation. More information about the input data structures is shown in subsection 3.5.1;
 - **Feature Analysis and Computation:** The propounded classification method is feature-based, thus it is necessary to first select adequate features to the data in observation. This procedure is followed by the analysis and computation of the selected features to the given Dataset. A detailed explanation is provided in subsection 3.5.2;
 - **Classifier:** An object to associate an input item to a specific class label is required. This object is the classifier. Labels are attributed based on the values of the designated features for the input Dataset, when compared to the Ground Truth subsets. Thorough information of the used classifier is displayed in subsection 3.5.3;
 - **Classification Function:** The purpose of the classification function is to regularise the results provided by the classifier. Nevertheless, this process can be computationally demanding meaning in some exceedingly large datasets it might be better to perform raw classification, without regularisation. This topic will be further discussed in subsection 3.5.4.

- **Post Processing:** This module provides the post processing steps, explained in section 3.6, to optimise the previously obtained classifications results. Optimization can either be done manually, by increasing the ground truth information, or automated. This work will emphasise automated methods, namely:
 - **Filtering:** The purpose of filtering is to analyse the incorrectly classified objects and to remove undesired points from a specific class label. Further details about this procedure are shown in subsection 3.6.1;
 - **Segmentation:** The objective of segmentation is to partition the classified objects into homogeneous clusters, to separate dissimilar objects. This process is discussed thoroughly in subsection 3.6.2.

3.3 External Resources

This section, as previously described, presents a brief description of the external libraries and software frameworks used during the development of this work.

3.3.1 Robot Operating System (ROS)

Robot Operating System (ROS) is a compendium of frameworks for robotics software development, providing several tools, libraries and services, designed to be hardware independent, provide inter-process communication, provide low-level device control and package management. The purpose of ROS is to provide an open-source collaborative robot software development environment, encompassing expertise from different research fields [39].

ROS-based processes, otherwise called nodes, perform computations in a graph architecture connected by edges called topics, like: sending or receiving sensorial information, state estimation, controller interfaces - to name just the most important for this work [39].

Besides the core functionalities of ROS, several tools are also provided in the framework. The most important used during the development of this work being tools to visualise and record data, namely rviz and rosbag. Rviz is a highly configurable three-dimensional visualisation tool for ROS where robots, their working environments and sensorial data, can be visualised. Rosbag "is a command line tool used to record and playback ROS message data" [39]. This tool has the capability to log ROS messages, by subscribing to the topics whose messages the user intends to record, saving them in a bag file, a rosbag specific format. The inverse operation is also available. In other words, playing messages from a bag file like the original nodes would have produced, which can be, for example, visualised using a visualisation tool like rviz [39].

ROS regularly releases a new distribution, which means older releases might not be compatible with certain packages. In this work, the distribution used was ROS Melodic Morenia, released on May 23, 2018, whose End of Life (EOL) date will be May 30, 2023 [39].

3.3.2 Point Cloud Library (PCL)

The increasing demand for libraries to process point cloud data led to the creation of Point Cloud Library.

The Point Cloud Library (PCL) is an open-source, cross-platform, library of algorithms, written in the C++ programming language, for two and three-dimensional image and point cloud processing. This library includes novel algorithms for "filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation. These algorithms can be used, for example, to filter outliers from noisy data, stitch 3D point clouds together, segment relevant parts of a scene, extract keypoints and compute descriptors to recognize objects in the world based on their geometric appearance, and create surfaces from point clouds and visualize them" [40].

3.3.3 Aerial Informatics and Robotics Simulation (AirSim)

AirSim is an open-source simulator for drones, cars and other vehicles, built on Epic Games' Unreal Engine, developed by Microsoft. The objective of this simulator is to provide "a platform for AI research to experiment with deep learning, computer vision and reinforcement learning algorithms for autonomous vehicles. For this purpose, AirSim also exposes APIs to retrieve data and control vehicles in a platform independent way" [1]. Through the usage of said APIs, it is possible to have a complete scrutiny of what data to collect and how to collect it, by interacting with the vehicle programmatically.

3.3.4 Octomap

Octomap is a 3D mapping framework with data structures and mapping algorithms. The 3D mapping implementation utilises an octree, a tree data structure "often used to partition a three-dimensional space by recursively subdividing it into eight octants" [30].

The map implementation provides the following advantages: it can generate a model of the entire environment without any previous conjecture; it can be updated in real time which means it can be used to map dynamically changing environments; the map expands whenever it is required to do so, without previous knowledge of the proportions of the environment; lastly, data is compressed, making it more practical and reducing memory limitations [17].

3.3.5 MeshLab

MeshLab is an open-source software for 3D mesh processing, which provides several other tools, like: "editing, cleaning, healing, inspecting, rendering, texturing and converting meshes. It offers features for processing raw data produced by 3D digitization tools/devices and for preparing models for 3D printing" [9].

The cleaning tools provided by this software comprise equivalent or unreferenced vertice removal, erasure of null vertices, faces or edges. Remeshing tools include surface

reconstruction algorithms based on ball-pivoting and Poisson reconstruction methods, mesh simplification based on quadratic error measurements, as well as surface subpartitioning. Noise removal tools include smoothing filters and curvature and visualisation analysis. Meshlab provides multiple range map registration based on the iterative closest point algorithm [9].

There is also "an interactive direct paint-on-mesh system that allows users to interactively change the color of a mesh, to define selections and to directly smooth out noise and small features" [9].

3.3.6 Computational Geometry Algorithms Library (CGAL)

The Computational Geometry Algorithms Library (CGAL) "is an open source software library of computational geometry algorithms" [37]. This library is fundamentally written in C++, with some packages providing support to the Java and Python programming languages as well. This project is the result of the work done by a consortium of academic institutions whose objective is to distribute efficient and effortless computational geometry algorithms, to be used "in various areas needing geometric computation, such as geographic information systems, computer aided design, molecular biology, medical imaging, computer graphics, and robotics" [37].

This library provides several packages with geometric algorithms. However, the one that was mainly used during the development work is the Classification package [13]. The classification package, developed by Simon Giraudot and Florent Lafarge [13] provides classification algorithms for different data kinds, which support several features and labels. Furthermore, these algorithms are implemented to be flexible with local own features defined by the users, as well as, custom input data with user defined labels.

3.4 Data Acquisition

In 3D computer graphics, three-dimensional data is generally represented as follows: point cloud data, polygon meshes, volumetric representation or projected depth image views [38]. A polygon mesh, in three-dimensional computer graphics, is a collection of vertices, edges and faces which compose a three-dimensional object. Generally, these faces comprise triangles, quadrilaterals or other convex polygons. This representation is particularly interesting since it provides surface irregularities which can be useful for mesh analysis [16]. A voxel is a volumetric representation of a value within a regular grid in space. In this representation, the position of voxels is obtained by inference in relation to the position of other voxels [29]. The projected depth image views, or depth map, in computer graphics, are images or channels which possess significant information about the relative distance of a certain viewpoint to the surface of an object. Single or multi-view depth maps can be retrieved with 3D scanners and utilised for the reconstruction of 3D shapes [27].

The point cloud representation is a set of points in space representing a 3D shape, usually obtained with a 3D scanner or the usage of photogrammetric techniques. This representation is similar to raw sensor data. It is also canonical, or to put it another way, point cloud data can be effortlessly obtained through the conversion of other representations. Similarly, the opposite conversion is also possible, generally called surface reconstruction. Generally point cloud data is converted before being fed to deep learning procedures. However, recently, feature learning on raw, unordered point cloud data has been studied [38]. This work will focus on instance and semantic segmentation (otherwise known as classification) of raw point cloud data.

Furthermore, data, specifically point cloud data in this work, can be both real or synthetic. Real data, as the name implies, is captured in a real life environment using a 3D scanner. The raw data can then be processed with computer vision algorithms. Synthetic data, on the other hand, is generated computationally with specific 3D graphic software. "In theory, objects in the synthetic datasets are complete, without any occlusion and background. In contrast, objects in the real-world datasets are occluded at different levels and some objects are contaminated with background noise" [14]. Furthermore, synthetic data is generally less expensive and the user has full scrutiny over what to capture and how to do it. For these reasons, synthetic data can be very useful for algorithm development and testing in simulated virtual environments. Nevertheless, translating the results of said algorithms to real life data can, sometimes, be a non-trivial task and produce undesired results so it is a good practice to assure the methods produced are adequate to real life environments.

In this work, the proposed methods will be tested for both synthetic and real life data. A detailed explanation of the data acquisition processes will be presented in the following subsections.

3.4.1 Synthetic Data

Originally, the data was intended to be acquired using ALS on-board a UAV platform in a real-life environment. However, due to the COVID-19 pandemic, academic institutions had imposed on them strict public health and social measures which hindered this process.

In order to solve this issue, a different approach was taken which consisted in the usage of simulated LiDAR data in a virtual environment. The simulator used for this purpose was AirSim. After collecting the simulated data, this data needs to be processed in order to be used by the ROS Network. This processing is achieved with Octomap. A sequence diagram illustrating the data acquisition process will be presented, in **Figure 3.2**, followed by a thorough explanation of the process.

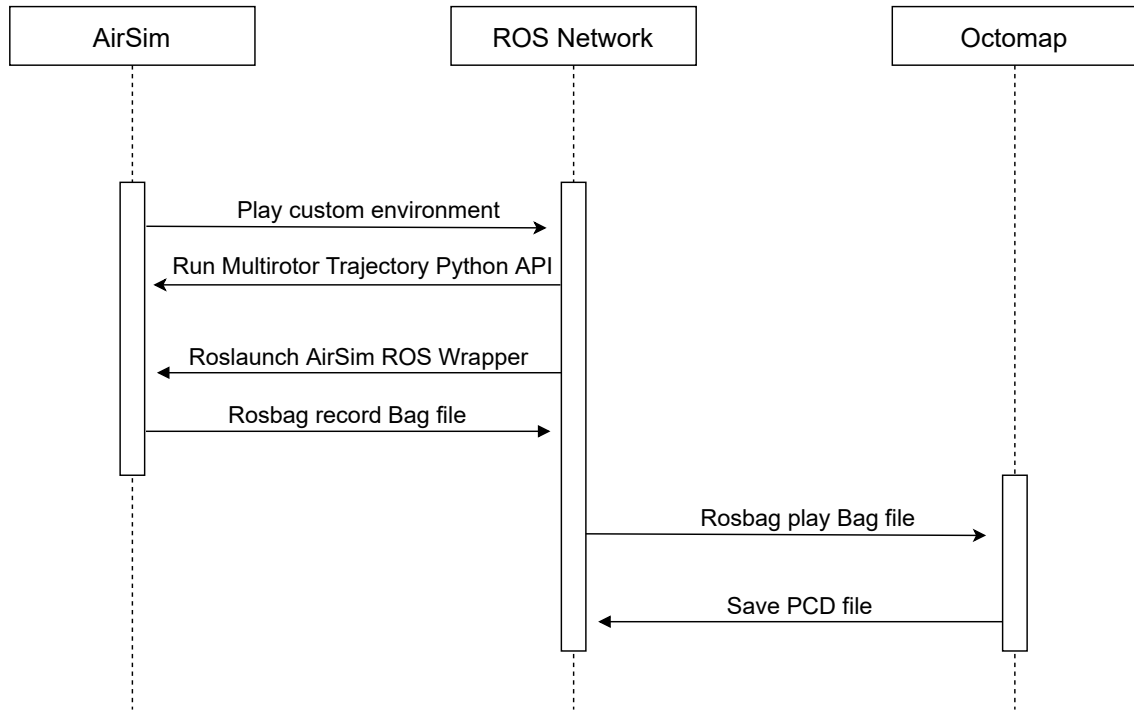


Figure 3.2: Sequence diagram of Synthetic Data Acquisition process. First, LiDAR data is captured using AirSim. Data is then published to the ROS network and, via sensor fusion, mapped with Octomap to a point cloud data format.

AirSim provides support to multiple sensors. However, the one that will be primarily focused is LiDAR. In order to be able to utilise the range of sensors available and more accurately emulate a real sensor, in this particular situation Velodyne’s VLP-16 sensor, a JSON settings file must be configured. These settings include the number of channels/number of lasers of the LiDAR, the number of points captured each second, the number of rotations each second, the start and end horizontal Field of View (FoV), the upper and lower vertical FoV, the position of the sensor in relation to the multirotor (in North-East-Down (NED) coordinates), and the orientation of the LiDAR in relation to the multirotor (in degrees, in relation to front vector +X). The values of these settings, following Velodyne’s VLP-16 sensor datasheet [44] are presented in the following bulleted list.

- **Number of channels/lasers:** 16;
- **Points per second:** 300.000;
- **Rotations per second:** 10;
- **Horizontal FoV Start:** 0;
- **Horizontal FoV End:** 359;
- **Vertical FoV Upper:** -15

- **Vertical FoV Lower:** -45;
- **Sensor position (in relation to the multirotor):** $X = 0, Y = 0, Z = -1$;
- **Sensor orientation (in relation to the multirotor):** Roll = 0, Pitch = 0, Yaw = 0.

Despite the LiDAR setting configuration, there is still a concern: to acquire the LiDAR data and publish it over the ROS network. For this purpose, there is an AirSim ROS Wrapper module which makes the integration between the simulator and the ROS network, containing two ROS nodes: a wrapper over the simulator's multirotor C++ library and a proportional-derivative (PD) position controller. This way, it is possible to subscribe to the topic publishing LiDAR data so it can be recorded using the rosbag package to be posteriorly used by other applications.

As previously mentioned, the goal is to gather simulated LiDAR data in a simulated environment. The simulator provides a few environments, however, it is also possible to setup a custom environment. For this purpose, a simple environment consisting of royalty-free assets of multiple trees of three different species (110 pine trees, 14 scots pine trees and 24 fir trees) and a power line structure with 8 pylons whose insulators are connected by 8 power lines was created using Epic Games' Unreal Engine 4.24. A screenshot of said environment is presented in the following figure, **Figure 3.3**.



Figure 3.3: Screenshot of the Synthetic Data simulated environment, captured in Unreal Engine 4.

The multirotor traverses the simulated environment as close to the power line structure and the vegetation as possible, through a predefined scripted trajectory given by a customised Python API. This API starts by establishing the connection to the simulator, enabling the programmatic control over the multirotor. Then, it waits for a key press for

takeoff and for each movement instruction. Lastly, after another key press, it returns to its original state. Simultaneously, through the integration between the simulator and ROS, using the previously mentioned ROS Wrapper module, the LiDAR captures the point cloud data which is then published to a ROS topic and visualised using rviz.

LiDAR data is published over the ROS network. However, through sensor fusion and mapping algorithms it is possible to obtain a state estimation, and, consequently, reshape data to a more practical format. For this purpose, OctoMap was utilised.

Using the `octomap_server` package it is possible to progressively create the 3D map of the environment, from incoming LiDAR data, and compress it to a ROS compatible format that can be used by other nodes. In this particular situation, the format of interest is the Point Cloud Data (PCD) file format. The PCD file format is the native file format in PCL, which complements existing 3D point cloud data file formats. A PCD file is divided in two segments: header and data.

The header contains the PCD file version; the dimensions/fields each point can have; the dimension size in bytes and its type as a char, be it signed, unsigned or float type; each dimension element count; the width of the dataset which can be the total number of points or number of points in each row, for unorganised and organised datasets respectively; the height of the dataset, which can specify the total number of rows or be set to 1, for organised and unorganised datasets respectively; the viewpoint of the point cloud, which provides information for future transforms and is described as a "translation (tx ty tz) + quaternion (qw qx qy qz)" [40]; the total amount of points in the point cloud; and the data type of the dataset, be it ascii or binary. Data, as the name suggests, contains the Cartesian coordinates values of the point cloud data.

Despite being the native file format in PCL, this file format provides several advantages over other file formats, the most important being: the different data types it supports; the capability to process organised point clouds; reading and writing data to disk is faster; it provides n-D histograms for feature descriptors which can be very useful for 3D perception [40].

The resulting PCD file from the mapping of the simulated environment using the incoming LiDAR data is shown in the following figure, **Figure 3.4**.

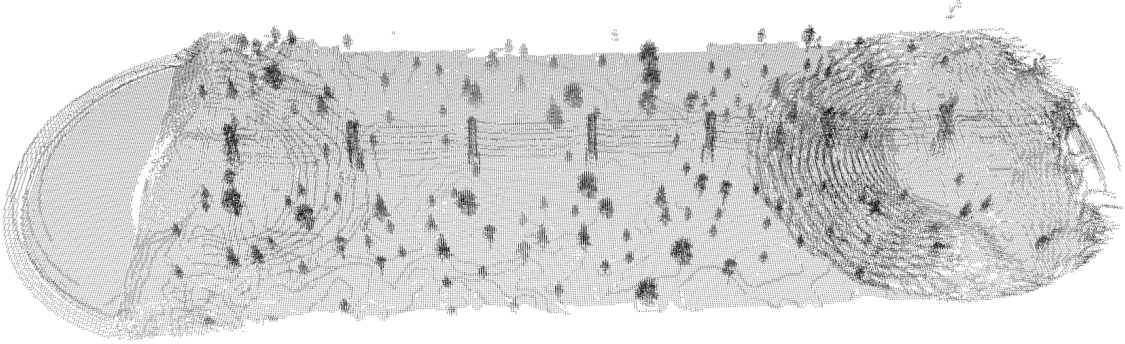


Figure 3.4: Screenshot of the resulting Point Cloud Data (PCD) file after mapping the simulated environment.

3.4.2 Real Data

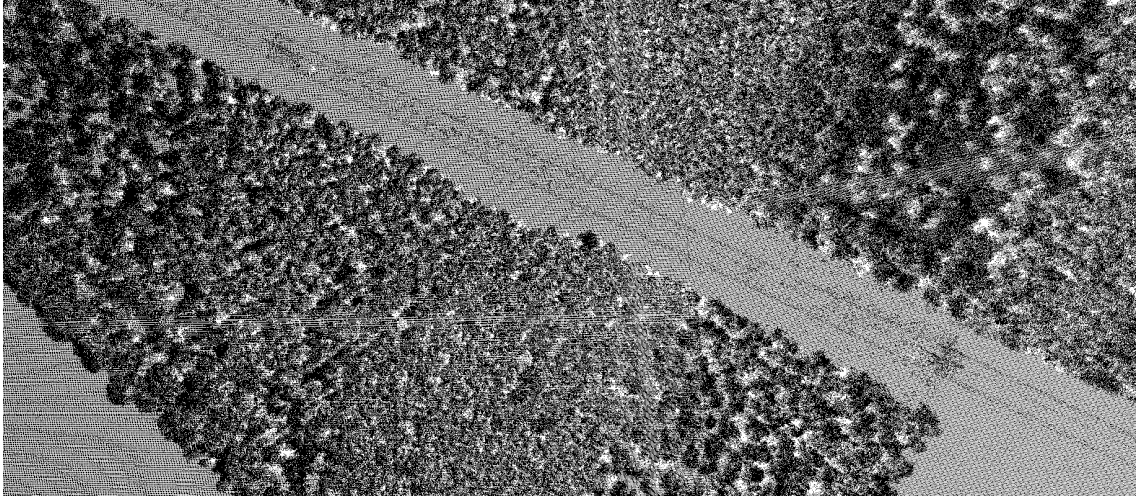
As previously mentioned, the COVID-19 pandemic negatively impacted the data acquisition process in a real life environment. Nonetheless, in the latest years several datasets openly accessible to the public have been published in repositories (mostly by universities and organisations) which further fostered point cloud processing researches [3, 14].

One of these repositories is Open.NRW, an open information and data platform from the North Rhine-Westphalia state, in Germany. This platform provides federal government and municipalities a technical infrastructure to publish open data [35].

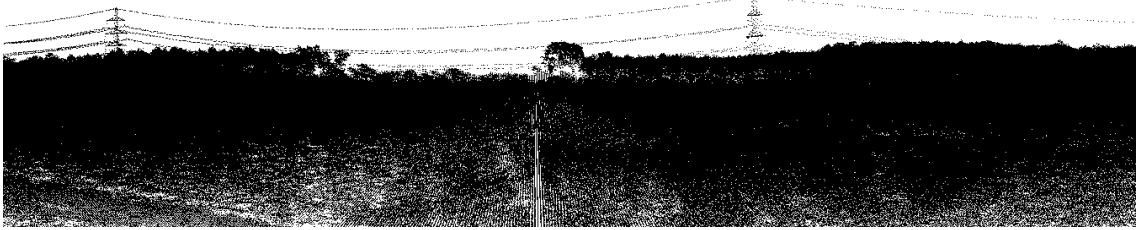
The data provided encompasses several scientific and social-economic sectors, however, the major focus for this work is LiDAR data captured in this state. Hundreds of files from both urban and non-urban environments are available, including environments with the presence of several vegetation elements like small bushes or full-fledged trees and electrical structures such as transmission towers and electrical wires.

The file format of these data files is the LAZ format which compresses large amounts of LiDAR data in a, relatively, small file. Usually it results from the compression of the most commonly used LAS format. "The LAS file format is a public file format for the interchange of 3-dimensional point cloud data data between data users" [23]. This file format was created as an alternative to proprietary file formats, which cannot be used in certain working environments or systems, as well as ASCII file format, which, despite being generic possesses certain performance and information loss limitations. The LAS file format is not only generic, but it can also keep relevant LiDAR information while not being too convoluted [23].

In this work specifically, the data files are converted to a different file format, in order to be processed by the previously mentioned tools. This process will be later explained in this dissertation. One data file sample, containing vegetation, electrical structures and ground points is presented in the next figure, **Figure 3.5**.



(a) The top view where one can identify Ground and Vegetation points. Pylon and Line points are also present in the figure, even though not as contrasting.



(b) The side view where Pylons and Lines are more visible. Vegetation and Ground points are also visible, despite not being possible to discern from each other.

Figure 3.5: Close-ups of the real life LiDAR sample Dataset from Open.NRW.

3.4.3 Overview

For posterior processing, the datasets presented in the previous subsections were manually labelled according to the classes of items in study. A thorough overview of the aforementioned datasets is shown in the following table, **Table 3.1**, which includes the total number of points and the number of points for each item class.

Table 3.1: Detailed information of manually labelled point cloud datasets, where synthetic data is represented by S# and real data is represented by R#.

| Sets | Total #Points | Ground | Vegetation | Pylons | Lines |
|------|---------------|-----------|------------|--------|-------|
| S1 | 135,651 | 104,623 | 24,225 | 4,500 | 2,303 |
| R1 | 2,782,469 | 1,381,214 | 1,390,493 | 2,589 | 8,173 |
| R2 | 5,799,533 | 2,946,336 | 2,841,415 | 2,272 | 9,510 |
| R3 | 4,765,377 | 1,933,420 | 2,823,869 | 1,397 | 6,691 |

3.5 Point Cloud Classification

In recent years, deep learning and its applications have been under scrutiny in many researches related to computer vision and data processing, namely point cloud data. Several methods have been proposed and datasets openly accessible to the public have been published (mostly by universities and industries) which further fostered deep learning for point cloud processing researches, including point cloud classification [3, 14]. A more detailed exposition of the publicly available datasets and deep learning methods for point cloud processing is available in the works of Guo et al. (2020) [14] and Bello et al. (2020) [3].

Nevertheless, point cloud processing using deep learning still faces certain challenges, like: datasets with small scales; irregularities in datasets, in other words, uneven sampling of points among dissimilar objects/regions; unstructured and unorganised datasets [3, 14]. These issues can be particularly challenging for Convolutional Neural Networks (CNNs), since CNNs perform convolutional operations which are negatively impacted by unstructured and unordered data [3]. A comprehensive description of these challenges is also provided in the aforementioned works, [14] and [3].

As mentioned in the previous chapter, this work utilises a synthetically generated dataset and real life datasets from an open data repository. The next step in the proposed method is to classify data according to their respective class labels. During the development of this work, the CGAL classification package was utilised to perform point cloud classification [13]. The next subsections will thoroughly explain this process.

3.5.1 Input Data Structures

As previously mentioned, CGAL was used to perform point cloud data classification. In CGAL the classification algorithms require a few input data structures, namely: Input Dataset, Label Set and Ground truth Subsets.

3.5.1.1 Input Dataset

The Input Dataset consists of two point cloud datasets, real and synthetic, previously presented in the Data Acquisition section of the present chapter. However, before being fed to the CGAL classification package, one last step must be taken into consideration: converting the input data to a format the classifier can understand. In this case, this format is the PLY polygon file format which describes a graphical object as an assembly of polygons. "The PLY format describes an object as a collection of vertices, faces and other elements, along with properties such as color and normal direction that can be attached to these elements. A PLY file contains the description of exactly one object. Sources of such objects include: hand-digitized objects, polygon objects from modeling programs, range data, triangles from marching cubes (isosurfaces from volume data), terrain data, radiosity models" [43]. This representation can be particularly useful since it can store

information from range measurements with reduced complexity and it is an extensible format. To put it another way, adding a label to a specified point pertaining to an object is as simple as adding a new vertex whose value is the index of said label [13].

3.5.1.2 Label Set

The Label Set contains the user-defined set of labels of the objects present in the Input Dataset. A label is no more than a representation of how an object should be classified. In CGAL, labels contain several properties: a name, which is usually the name of the object in common sense. A label set index, in other words the index of said label in an array containing all labels. A standard index which contains the value that is attributed to the appended vertex during the classification process. Lastly, a RGB color used for visualisation purposes. Nevertheless, neither the name, the standard index or the color are unique. What truly identifies a label is a label handler which is unique to the specified label. The name of the label can, however, be used to deduce other information about said label, when the label initialisation is done using only the label name [13]. In this work, the defined labels, which correspond to the objects in interest, are: Vegetation, Pylons and Electrical Wires.

3.5.1.3 Ground Truth Subset

The Ground Truth Subsets contain certain sets of labeled data which define an object recognition model "including the count, location, and relationships of key features" [22]. Labelling data is a process that can be done manually through visual interpretation or programmatically through data analysis, according to the intricacy of the problem [22]. "The collection of labels, such as interest points, corners, feature descriptors, shapes, and histograms, form a model. A model may be trained using a variety of machine learning methods. At run-time, the detected features are fed into a classifier to measure the correspondence between detected features and modeled features" [22]. Further information about Ground Truth Data is presented in the work of Krig (2014) [22]. For this work, ground truth subsets were obtained through manual annotations, based on the visual interpretation of the input datasets.

3.5.2 Feature Analysis

"In machine learning and pattern recognition, a feature is an individual measurable property or characteristic of a phenomenon being observed" [7]. Generally, features are represented by numerical values, however, in pattern recognition sometimes strings and graphs are also used. Initially, the raw feature set can be too sizeable to manage and may have expendable features. Consequently, two crucial preliminary steps are generally performed when computing features in machine learning: feature selection and feature extraction. This process can then be automated using feature learning [7].

3.5.2.1 Feature Selection

Feature selection is a process of selecting descriptive, discerning and independent features to be used in machine learning model construction. The importance of selecting relevant features is that it facilitates the model interpretation of other researchers, it can significantly reduce training duration, it is easier to analyse and organise high-dimensional data, and it also decreases overfitting (to put it another way, when a model memorises a large number of examples in lieu of learning from features). The focal point of this process is to find and filter any redundant or irrelevant features present in data while minimising the information loss. Redundancy and irrelevance are independent, since any given relevant feature can be redundant when compared to another correlated relevant feature [11, 18]. As a separate assignment, feature selection can be supervised or unsupervised, or a combination of both techniques. However, "some supervised algorithms already have built-in feature selection, such as Regularized Regression and Random Forests" [11].

3.5.2.2 Feature Extraction

Feature extraction obtains relevant and non-redundant features from an initial set of measured data, generally used to describe large datasets. This process is similar to feature selection, however, feature extraction creates a new subset of features while feature selection retains a subset of the original features. "As with feature selection, some algorithms already have built-in feature extraction" [11]. Likewise, feature extraction can also be supervised or unsupervised [11].

3.5.2.3 Feature Learning

Feature Learning is a process that supplants non-automatic feature engineering and it consists of a collection of methods used to automatically perform feature detection or classification. The motivation that leads to this process is that certain assignments, like classification, generally need an input that is programmatically advantageous to process. Feature learning, like feature selection and extraction, can also be supervised (with labeled input data) or unsupervised (with unlabeled input data) [4].

3.5.2.4 Feature Computation

In CGAL, "features are defined as scalar fields that associate each input item with a specific value" [13]. Due to performance issues, namely memory consumption and processing time, the scalar value of said features is a floating-point number (simply called a float in computer science). Likewise, feature computation is done using a reduced number of scales, in this particular work five different scales. Features are first computed then added to a feature set. Furthermore, each feature is represented by a name and a feature handler. CGAL supplies a set of predefined features which are generally inclusive, however, it

is possible to define custom features, particularly if the input dataset contains certain characteristics that were not foreseen by said predefined features [13].

In this work, the selected features for the classification process are:

- **Eigenvalue:** Computes one of the three eigenvalues in a local neighborhood. Eigenvectors and eigenvalues exist in pairs and determine the variance of certain data in a specified direction. That direction is given by the eigenvector while the value of data variance in that direction is given by a number, called eigenvalue [13];
- **Distance to Plane:** Calculates the distance between a point and a locally estimated plane. This feature can be particularly interesting to characterise non-planar input data, like Vegetation or Pylons [13];
- **Vertical Dispersion:** Estimates the vertical dispersion of a point set within a local Z-cylinder around the points. In other words, how vertically noisy the specified point set is. The environments under inspection can be deconstructed into groups of 2D regions with dissimilar heights. For Pylons and Lines, these heights are constant, however, for Vegetation these heights can be very unsteady [13];
- **Elevation:** Calculates the distance of a local neighborhood to an estimation of the ground. This feature can be particularly helpful to differentiate ground points from power line points [13];
- **Height Below:** Measures the distance between a specific point and the lowest point in a local neighborhood. Similarly to the previous feature, this feature is also based on the local elevation [13];
- **Height Above:** Identical to the previous feature. However, in this situation, the distance between a specified point and the highest point in a local neighborhood is calculated instead [13];
- **Vertical Range:** Yet another feature based on local elevation, which computes the distance between the highest and lowest points in a local neighborhood [13].

As previously stated, in CGAL users can define their own features, however, the algorithms are optimised to use the provided predefined features. In theory, the most predefined features used, the better the results. For this reason, CGAL provides a module which makes an estimation of minimum pertinent scale for the features used, creates the required analysis structures and it utilises all the predefined features according to the accessible property maps [13].

3.5.3 Classifier

As previously described, classification using CGAL is achieved through feature computation and can be used to classify "point sets, surface meshes and clusters" [13].

"Classification relies on a classifier: this classifier is an object that, from the set of values taken by the features at an input item, computes the probability that an input item belongs to one label or another" [13].

A classifier model in CGAL receives the index of an input item and returns the probability of said item being contained in each label in the form of a vector. It then returns a numerical value of one for a label and item pair if said item belongs to the specified label or a numerical value of zero otherwise [13].

CGAL supplies different classifiers, however, the one that is mentioned to be the best and advised by the developers, which is also the one that was utilised during this work, is the **ETHZ Random Forest classifier** [47], "a classifier based on the Random Forest Template Library developed by Stefan Walk at ETH Zurich" [13] provided under the MIT license without further software installations (besides the respective CGAL distribution). As previously mentioned, this classifier needs a ground truth training subset and the training algorithm not only requires a large number of inliers, but it also consumes more memory during runtime and has larger configuration files than the other classifiers provided. On the other hand, it is faster and produces remarkably better results, namely as the number of labels increases [13].

3.5.3.1 Random Forest

Random forest algorithms are widely used in machine learning, namely in classification, due to their simplicity and flexibility. Furthermore, these algorithms generally generate reasonable results without hyper-parameter fine tuning [12].

Before explaining how the Random Forest model works, it is important to understand its constitutive blocks: decision trees. A decision tree is composed of multiple nodes which, based on certain features, fracture data under analysis in different groups, named branches, as dissimilar to each other as possible but whose resulting members, called leaf nodes, represent the same class label. Leaf nodes which do not have any children are simply called leafs [12].

"Random forest is a supervised learning algorithm" [12] that, analogously to a real forest, is an ensemble of trees (in this situation however, decision trees). The presupposition of the Random Forest algorithm is that creating compact decision-trees with a limited set of features is computationally inexpensive [6]. The output of these decision trees is then amalgamated to produce a more accurate prediction. It is called a Random Forest because in lieu of splitting nodes using the most relevant feature found the algorithm utilises the best feature from a random subset of features, typically resulting in a better model [12]. An additional characteristic of this algorithm is the ability to compute the relative relevance of each feature in a prediction. This information can be very valuable

to decide which features do not provide a positive contribution to the predictions and can be consequently excluded which in turn prevents the model from suffering overfitting, or in other words, having an excessive number of features for the given data [12].

As stated, this algorithm outputs reasonable results without hyper-parameter tuning, however, hyper-parameters can be used to improve prediction accuracy and the model speed [12]. On the other hand, achieving better predictions is the result of expanding the number of trees, which in turn slows the algorithm, thus meaning it might not be adequate for real-time predictions. Furthermore, "Random Forest is a predictive modeling tool and not a descriptive tool" [12]. In other words, it is not suitable to achieve a description of relationships present in data.

In practice, this algorithm works as described:

1. For each decision tree select a bootstrap sample from the training set, "Bootstrap Sampling is a method that involves drawing of sample data repeatedly with replacement from a data source to estimate a population parameter" [20]. Bootstrap Sampling can be applied to ensemble machine learning algorithms, like Random Forest, to circumvent overfitting issues and reduce computational expenses [20]. In this case, the training set is the ground truth subset. The outcome of a decision tree is then attained with a modified learning algorithm;
2. The modified learning algorithm works like so: a random subset of features, among the total set of features, is firstly chosen. Then, each tree node is analysed for the specified subset of features. The node is split based on the most relevant feature from the random subset of features, instead of the initial set of features. Since this random subset of features is significantly reduced when compared to the initial set of features, it drastically decreases the learning time of the decision trees. "Deciding on which feature to split is oftentimes the most computationally expensive aspect of decision tree learning" [6].

For a better understanding of the implementation of this algorithm, its pseudocode is displayed in **Algorithm 1** which follows [6] meticulously.

As aforementioned, the Random Forest algorithm used during the development of this work uses as a basis the ETH Zurich Random Forest Library. Furthermore, two important parameters must be instantiated: the number of trees and the depth of the forest [13, 47].

- **Number of trees:** 25;
- **Depth of the forest:** 20.

Algorithm 1 Random Forest [6]

Inputs:Training Set, S Feature Set, F Number of Decision Trees in the Forest, N

```
1: procedure RANDOMFOREST( $S, F, N$ )
2:    $H \leftarrow \emptyset$ 
3:   for  $i \in 1, \dots, N$  do
4:      $S^i \leftarrow$  Bootstrap Sample from  $S$ 
5:      $h_i \leftarrow$  RANDOMIZEDTREELEARNING( $S^i, F$ )
6:      $H \leftarrow H \cup h_i$ 
7:   end for
8:   return  $H$ 
9: end procedure
10: procedure RANDOMIZEDTREELEARNING( $S, F$ )
11:   At each node:
12:      $f \leftarrow$  reduced subset of features from  $F$ 
13:     Split node according to the best feature in  $f$ 
14:   return LearntTree
15: end procedure
```

3.5.4 Classification Functions

"Classification is performed by minimizing an energy over the input dataset that may include regularization" [13]. CGAL supplies three different classification methods, with differing speed and output quality:

- **Raw Classification:** This is the quicker of the methods provided, however, it presents the noisiest results. This method minimises the energy over the input data by summing the itemwise energies obtained with the classifier, without regularisation [13];
- **Local Regularisation:** This method provides a compromise between speed and quality of the output results. It performs local regularisation by computing the energy of all items in a local neighborhood, which outputs less noisy results. The quality of the output results increases as the local neighborhood is expanded, however, it also increases computation times [13];
- **Global Regularisation:** This is the slowest method but also the one that provides the best results. "A graph cut based algorithm (alpha expansion) is used to quickly reach an approximate solution close to the global optimum of this energy" [13]. This method fractures the input data in piecewise constant clusters which enables the correction of sizeable incorrectly classified clusters [13].

As stated in CGAL documentation, for "a point set of 3 millions of points, the first

method takes about 4 seconds, the second about 40 seconds and the third about 2 minutes" [13]. In this work, the classification function utilised was Local Regularisation, since it provides a reasonable trade-off between the output result quality and computational resources consumed. Ideally, Global Regularisation should be used, to further improve result quality, however, for sizeable environments (with millions of points), this method can be computationally demanding.

3.5.5 Evaluation

In order to evaluate the performance of a machine learning model, certain metrics (named evaluation metrics) are used. There are several kinds of evaluation metrics to test a model, however, the ones that are oftenly used in 3D Classification are Precision and Recall.

Precision, for a specified class, is the "ratio of true positives over the total number of detected positives" [13]. In other words, precision is the number of relevant items retrieved. Its mathematical expression can be seen in the following equation, **Equation 3.1**.

$$Precision = \frac{TP}{TP + FP} \quad (3.1)$$

where TP is the number of true positive samples and FP is the number of false positive samples.

Recall, for a given class, is the "the ratio of true positives over the total number of provided inliers of this label" [13]. To put it another way, recall is the quantity of relevant items amidst all the retrieved items. The succeeding equation, **Equation 3.2**, presents its mathematical expression.

$$Recall = \frac{TP}{TP + FN} \quad (3.2)$$

where, once again, TP is the number of true positive samples and FN is the number of false negative samples.

Another commonly used metric, to evaluate the overall performance of the classifier, is the Overall Accuracy. This metric provides the probability of a singular object being classified as belonging to its real class. The mathematical expression of this metric is displayed in the next equation, **Equation 3.3**.

$$OverallAccuracy = \frac{TP + TN}{T} \quad (3.3)$$

where TP is, once more, the number of true positive samples, TN is the number of true negative samples and T is the total number of samples.

3.6 Post-Processing

Regardless of the classifier utilised, it is never possible to achieve flawless results during the classification process. For this reason, some post-processing operations might be required to optimise the classification results. This process can be either manual or automated. Generally speaking, manual post-processing can be either done by successively increasing the Ground Truth Subsets followed by a reclassification of the Input Dataset, or by utilising specific software that can process and edit the labelled data. These methods can be effective for smaller, similar, environments, but they can be very time consuming for multiple, dissimilar, large environments. Furthermore, they require a human operator to either manually label more data to increase the training set of the classifier or to manually correct the wrong classifications. For this reason, this work will focus on automated post-processing methodologies which, in theory, should be more generalist and should not be so time consuming.

In order to automate the optimisation of the classification results, it is first necessary to identify the issues faced by the classifier. To summarise, the following incorrect predictions can be expected to be done by the classifier:

- Ground points being incorrectly classified as other object classes. It is possible to tackle this issue following the assumption that ground points possess similar characteristics to a planar surface while the other objects in observation do not;
- Low Vegetation incorrectly classified as Lines or Pylons. In the first case, this issue should be particularly straightforward, taking into account the objects in observation have dissimilar elevations. However, in the second case, depending on the distance between the Vegetation and the Pylons, this could be the most challenging issue to overcome. When the distance is sizeable enough, it should be possible to separate different objects using a segmentation algorithm. Troubles arise when this distance is not significant to the segmentation algorithm. In certain situations this issue could even be troublesome for manual corrections;
- Tall Vegetation misclassified as Lines or Pylons. Similar to the previous topic, the difficulty to overcome this issue is intrinsically related to the distance between said objects. Furthermore, in this situation, it is not possible to separate Vegetation from Lines based on elevation criteria (like the previous solution for Low Vegetation);
- Line spans being wrongly classified as Pylons. The least problematic kind of incorrect predictions, since the main focus is to separate Vegetation from Lines and Pylons. Nevertheless, one possible solution for this issue is to use a model fitting algorithm, namely line/catenary curve fitting.

To perform automated classification post-processing, essentially two kinds of procedures were executed: filtering and segmentation. These procedures will be thoroughly explained in the following subsections.

3.6.1 Filtering

Point cloud filtering, as the name implies, is a process to remove undesired points from data. Generally, filtering is performed to pre-process point cloud data, which "inevitably suffers from noise contamination and contains outliers, due to the limitations of sensors, the inherent noise of the acquisition device, the lighting or reflective nature of the surface or artifact in the scene" [15]. Nevertheless, filtering can be similarly done after classification, in order to optimise results by rectifying incorrect predictions.

Filtering algorithms are usually separated into seven categories: statistical-based, neighborhood-based, projection-based, signal processing based, Partial Differential Equations (PDEs), hybrid techniques and other methods not framed in these categories [15].

Statistical-based filtering techniques make use of different statistical concepts, depending on the most suitable approach for the point cloud data at hand. These statistical concepts can be probability distributions, likelihood functions, least squares, sparse matrices - to name a few [15].

Neighborhood-base filtering techniques utilise similarity criteria to ascertain the position of a filtered point in relation to its neighborhood. The similarity criteria can be determined by the regions the points are contained in, their positions or normal values [15].

Projection-based filtering techniques "adjust the position of each point in a point cloud via different projection strategies to filter point cloud [15]".

Signal processing based methods make use of different signal processing concepts, like Laplacian operators, Fourier transforms, Wiener filters, for example [15].

Partial Differential Equations (PDEs) are widely used in computer vision and have been proven to accomplish mesh and image denoising with favorable results. This technique can be extensively used to filter point cloud data as well. However, these techniques can take a lot of time to process point clouds [15].

Hybrid filtering techniques combine multiple point cloud filtering techniques to filter the raw data [15].

Besides these methods there are other methods not framed in these categories. A more detailed explanation of all of these methods as well as a succinct comparison of their performance, benefits and limitations can be found in the work of Han et al. (2017) [15].

In particular, the objective of the filtering process in this work, as mentioned before, is to optimise classification results. This goal was achieved using the PCL outlier removal and noise filtering library. The filtering techniques utilised in this work were passthrough filters [40].

Passthrough filters can be efficient to isolate different groups of points with dissimilar coordinates in a specified axis. It can be used to separate low Vegetation points from Line points, for example. In order to implement the passthrough filters, using PCL [40], the subsequent steps are necessary:

1. Define the filter field name, which coordinate axis contains the points that are intended to be filtered;
2. Delineate the filter limit, the interval of accepted values within the previously specified coordinate axis;
3. Alternatively, the filter limit indicated before can also be the interval of rejected values by performing the negation of the filter limits.

3.6.2 Segmentation

Point cloud segmentation, on the other hand, is a point cloud classification procedure that partitions data into several homogeneous regions, otherwise called clusters, according to certain similarities [33]. However, point cloud segmentation can be challenging. "The point cloud data are usually noisy, sparse, and unorganised. The sampling density of points is also typically uneven due to varying linear and angular rates of the scanner. In addition, the surface shape can be arbitrary with sharp features and there is no statistical distribution pattern in the data. Moreover, due to the limitations of the 3D sensors, the foreground is often highly entangled with the background" [33].

Point cloud segmentation methods are commonly divided into five different categories: "edge based methods, region based methods, attribute based methods, model based methods, and graph based methods" [33].

Edge based methods segment point cloud data according to object shape. Clusters are obtained from the analysis of the boundaries of several regions extracted from the point cloud data. One of the advantages of these methods is how fast the segmentation process can be, however, these methods are very susceptible to noisy data and uneven density of point cloud data, which are general occurrences when capturing data [33].

Region based methods utilise information from neighboring points with comparable characteristics to extract clusters. Consequentially, clusters with dissimilar characteristics are also identified and grouped in different clusters. These methods are more insensitive to noise than edge based methods. However, they present issues of over and under segmentation when a single object is perceived as different objects or when different objects are perceived as the same object, respectively [33, 34].

Attribute based methods segment point cloud data based on certain clustering attributes, however, the performance of these methods is dependent of the quality of the clustering attributes [33].

Model based methods cluster points based on their geometric primitive shapes, for example, a sphere, a cube, a pyramid, a plane, and so forth. The major advantage of these methods are their swiftness and robustness with outliers. On the other hand, multiple point cloud sources can present accuracy issues [33].

Graph based methods assume point cloud data like a graph. In this kind of method, vertexes are represented as points from data and the edges connect specific pairs of

neighboring points. In comparison to the other methods presented, graph based methods are more accurate when clustering complex scenes, including noisy data and data with uneven density. On the other hand, generally these methods cannot run in real time, which means, they might require offline training or specialised hardware [33].

Thorough descriptions of the previously mentioned methods are shown in the work of Nguyen and Le (2013) [33]. This work also presents a comparison of different segmentation methods, including their advantages and disadvantages.

In practice, the purpose of the segmentation process is to correct wrong predictions (from the classifier) by trying to separate objects with dissimilar characteristics in different clusters. A Progressive Morphological Filter is first used to isolate ground and non-ground points. An Euclidean Clustering algorithm was used to separate Vegetation points from Line and Pylon points. However, as mentioned before, when the distance between Vegetation and the other objects in observation is reduced the algorithm might not be able to discern individual objects correctly.

As stated previously, a progressive morphological filter is utilised to perform the segmentation of Ground points. "Mathematical morphology composes operations based on set theory to extract features from an image. Two fundamental operations, dilation and erosion, are commonly employed to enlarge (dilate) or reduce (erode) the size of features in binary images" [55]. The combination of these operations results in opening and closing operations. These mathematical concepts can be expanded to LiDAR data analysis, namely of continuous surfaces. Morphological filters can be used to segment Pylons and trees from LiDAR data, however, it can be challenging to segment all non-ground objects of dissimilar proportions with a fixed filtering window size. A progressive morphological filter overcomes this issue by moderately expanding the filtering window size [55].

How the progressive morphological filter works will be thoroughly explained in the next steps, meticulously following [55]:

1. Load input dataset and construct a minimum surface grid by determining the minimum elevation for every grid cell. These grid cells will contain point coordinates. However, if any given cell does not contain any measurements, it is allocated the value of the closest point;
2. An opening operation is employed to the surface grid. The inputs of the filter, during the first iteration, are the minimum elevation surface and a starting filtering window size. For the succeeding iterations, the inputs are the filtered surface from the prior iteration and an incremented window size from the following step. This step outputs the "further smoothed surface from the morphological filter" [55] and "the detected non-ground points based on the elevation difference threshold" [55].

There are essentially two methods to select the window size of this algorithm, linearly or exponentially. The window size can be increased linearly following the next equation, **Equation 3.4**:

$$w_k = 2kb + 1 \quad (3.4)$$

where w_k is the window size, k is a constant value which can range from 1 to M , and b is the starting window size. The maximum window size is given by $2Mb + 1$. This method "guarantees that the filter window is symmetric around the central point so that the programming of the opening operation is simplified" [55]. It also conserves topographic features, at the cost of a high processing time for areas with few ground points (in comparison to non-ground points).

On the other hand, the window size can be exponentially increased according to **Equation 3.5**:

$$w_k = 2b^k + 1 \quad (3.5)$$

where, once again, w_k is the window size, k is still a constant ranging from 1 to M , but b is now the base of an exponential function. The maximum window size is now given by $2b^M + 1$;

3. The filtering window size is incremented and "the elevation difference threshold is calculated" [55]. The previous step and this one are replicated "until the size of the filter window is greater than a predefined maximum value. This value is usually set to be slightly larger than the maximum building size." [55]

A progressive morphological filter can be either one or two-dimensional based on the shape of the filtering window. A two-dimensional filter has a geometric shaped window, such as a rectangle or a circle. Meanwhile, a one-dimensional filter has a window stipulated by a segment of a line, for example. Nevertheless, the algorithm for both a one and two-dimensional filter is analogous. For clarity, without loss of generality, the one-dimensional filter algorithm will be presented in **Algorithm 2** scrupulously following [55].

Algorithm 2 Progressive Morphological Filter [55]

Inputs:

Point Cloud Data, P
 Cell size, c
 Incremental window size parameter, from (3.4) or (3.5), b
 Maximum window wize, w_M
 Terrain slope, s
 Initial Elevation difference threshold, dh_0
 Maximum Elevation difference, dh_{max}

Output:

Two classified point sets containing ground and non-ground points

```

1: procedure PROGRESSIVEMORPHOLOGICALFILTER( $P, c, b, w_M, s, dh_0, dh_{max}$ )
2:   Calculate minimum and maximum x and y values
3:   Calculate number of rows ( $m$ ) and columns ( $n$ ) according to  $m = \text{floor}[(\max(y) - \min(y))/c] + 1$  and  $n = \text{floor}[(\max(x) - \min(x))/c] + 1$ 
4:   Create 2D array  $A[m, n]$  to save data points. The cell these points belong to is decided by their x and y coordinates. If multiple points belong to same cell, point with minimum elevation is chosen
5:   Interpolate elevation of cells without points from  $A$  using nearest neighbor method. Set value of x and y coordinates to zero. Copy  $A$  to  $B$ . Initialise elements of array  $\text{flag}[m, n]$  with 0.
6:   Ascertain series of  $w_k$  according to (3.4) or (3.5), where  $w_k \leq w_M$ 
7:    $dh_T = dh_0$ 
8:   for each window size  $\in w_k$  do
9:     for  $i = 1 \in m$  do
10:       $P = A[i, :]$  ▷  $A[i, :]$  is row of points at row  $i$  in  $A$  and  $P$  is a 1D array
11:       $Z \leftarrow P_i$  ▷  $P_i$  elevation values saved in 1D array  $Z$ 
12:       $Z_f = \text{EROSION}(Z; w_k)$ 
13:       $Z_f = \text{DILATION}(Z_f; w_k)$ 
14:       $P_i \leftarrow Z_f$  ▷  $z$  values from  $P_i$  replaced with  $Z_f$ 
15:       $A[i, :] = P$  ▷ filtered row of points  $P_i$  back to row  $i$  of array  $A$ 
16:      for  $j = 1 \in n$  do
17:        if  $Z[j] - Z_f[j] > dh_T$  then
18:           $\text{flag}[i, j] = w_k$ 
19:        end if
20:      end for
21:    end for
22:    if  $dh_T > dh_{max}$  then
23:       $dh_T = dh_{max}$ 
24:    else
25:       $dh_T = s(w_k - w_{k-1})c + dh_0$ 
26:    end if
27:  end for

```

```

28:   for  $i = 1 \in m$  do
29:     for  $j = 1 \in n$  do
30:       if  $B[i, j](x) > 0$  and  $B[i, j](y) > 0$  then
31:         if  $flag[i, j] = 0$  then
32:            $B[i, j]$  is a ground point
33:         else
34:            $B[i, j]$  is a non-ground point
35:         end if
36:       end if
37:     end for
38:   end for
39: end procedure
40:
41: procedure EROSION( $Z, w_k$ )
42:   for  $j = 1 \in n$  do
43:      $Z_f[j] = \min_{j-[w_k/2] \leq l \leq j+[w_k/2]}(Z[l])$ 
44:   end for
45:   return  $Z_f$ 
46: end procedure
47:
48: procedure DILATION( $Z; w_k$ )
49:   for  $j = 1 \in n$  do
50:      $Z_f[j] = \max_{j-[w_k/2] \leq l \leq j+[w_k/2]}(Z[l])$ 
51:   end for
52:   return  $Z_f$ 
53: end procedure

```

In the context of point cloud segmentation, Euclidean clustering is a method based on the Euclidean distance between neighboring points according to a specified distance threshold, as seen in **Equation 3.6**.

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2} \quad (3.6)$$

where p and q are the points whose distance is being calculated, $d(p, q)$. If this distance is lower than the specified threshold the points are considered to belong to the same cluster, otherwise they are considered to belong to different clusters [34, 40].

An important consideration about this method is how neighboring points are obtained. In this particular situation, a k-d tree was utilised. A k-d tree (otherwise named k-dimensional tree) "is a space-partitioning data structure for organising points in a k-dimensional space" [5]. In this kind of binary search trees, each leaf node corresponds to a k-dimensional point. Non-leaf nodes partition space into two halves, named half-spaces [5].

The implementation of the Euclidean clustering algorithm is achieved using PCL, according to the following steps [40]:

1. Generate a k-d tree structure for the incoming point cloud data;
2. Create an empty list for the different clusters and a queue for the points to be analysed;
3. For each point in the queue search the neighboring points and confirm whether that neighboring point has already been analysed or not. If not, add it to the queue. Once all the points inside the queue have been analysed, said queue is appended to the cluster list and reset;
4. Once the entire point cloud data has been processed and all clusters have been extracted, the algorithm terminates;
5. At the end of the clustering process, the clusters are saved in the disk as separate files for later analysis.

For a better interpretation of the algorithm implementation, the pseudocode for the said algorithm can be found in **Algorithm 3**, closely following [40].

Algorithm 3 Euclidean Clustering

Inputs:Point Cloud Data, P Distance Threshold, d_{th} **Initialise:**List of Clusters, $C \leftarrow \emptyset$ Queue, $Q \leftarrow \emptyset$ Point has NOT been processed, $notproc \leftarrow 1$

```

1: procedure EUCLIDEANCLUSTERING( $P, d_{th}, C, Q, notproc$ )
2:   for  $p_i \in P$  do
3:      $Q \leftarrow p_i$ 
4:     for  $p_i \in Q$  do
5:       search for the point set  $P_i^k$  of neighbors of  $p_i$  in a sphere of radius  $r < d_{th}$ 
6:       for  $p_i^k \in P_i^k$  do
7:         if point has already been processed then
8:            $notproc = 0$ 
9:         end if
10:        if  $notproc = 1$  then
11:           $Q \leftarrow p_i$ 
12:        end if
13:      end for
14:    end for
15:     $C \leftarrow Q$ 
16:     $Q \leftarrow \emptyset$ 
17:  end for
18: end procedure

```

RESULTS

In this chapter, the experimental results from the classification step are presented, followed by an analysis of said results. In the first section, **section 4.1**, the classification results are shown, presenting the respective evaluation metrics for both real and synthetic data. A brief description of the challenges faced during the classification process are also displayed. In the second section, **section 4.2**, the optimisation of the aforementioned classification results are shown.

4.1 Classification

As previously described, two types of data were used during the development of this work: real and synthetic data. The synthetic Dataset was captured with AirSim [1], by recording simulated LiDAR data through a predefined trajectory in a virtual environment. On the other hand, the real Datasets were captured in the North Rhine-Westphalia state, in Germany, and openly published by Open.NRW [35]. Ground truth subsets, as referred in the previous chapter, are manually labelled with the corresponding item class. The classes, also according to the previous chapter, are: Ground, Vegetation, Pylons and Lines. Thorough information of said ground truth subsets is provided in **Table 4.1**. This information includes the number of total points these subsets contain, the number of points comprised in each item class and the percentage of the dataset these points correspond to. In other words, the amount of points from the original dataset that were utilised to train the classifier. The higher this number, the more information the classifier attains during the training process and, consequently, the better the results. However, providing too much information defeats the purpose of using a classifier. For this reason, it is necessary to find a trade-off between the number of points that are used for training and the classification results.

Table 4.1: Detailed information of manually labelled ground truth subsets, where synthetic data is represented by S# and real data is represented by R#.

| Sets | #Points | Ground | Vegetation | Pylons | Lines | %Dataset |
|------|---------|--------|------------|--------|-------|----------|
| S1 | 2,499 | 1,275 | 579 | 503 | 142 | 1.842 |
| R1 | 9,550 | 3,093 | 5,657 | 421 | 379 | 0.343 |
| R2 | 54,396 | 21,579 | 31,710 | 493 | 614 | 0.937 |
| R3 | 74,961 | 15,815 | 58,317 | 449 | 380 | 1.573 |

For a better comprehension of the previous information a visual representation is provided in **Figure 4.1**. In particular, an illustration of the ground points from the Dataset R3, with different class labels represented by dissimilar colors for clarity (the color code itself is explained in the caption of the figure).

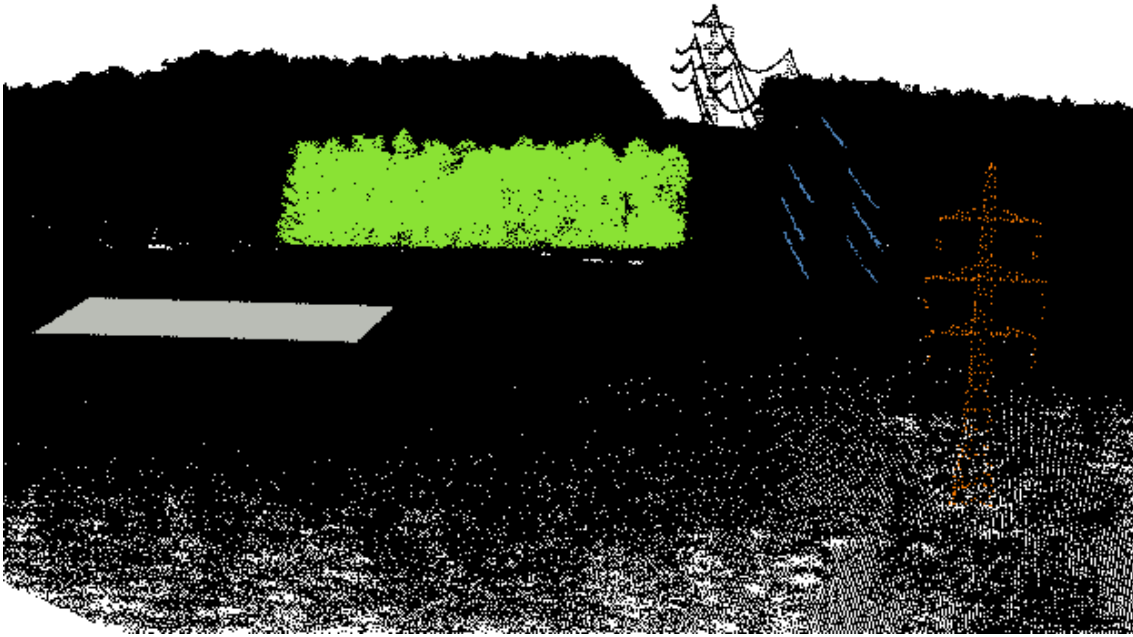


Figure 4.1: Ground Truth Subset from real life Dataset R3. Pylons are represented by the orange color, Lines are blue, Vegetation is green and Ground points are grey.

Once more, the classifier utilised in this work was the ETHZ Random Forest Classifier [47] and the classification function was Local Regularisation. One important note worth mentioning is that ground points were not taken into account for classification results. The reason for this being, for the problem this work intends to address, ground points do not provide valuable information, since the objective is to identify vegetation and electrical structures. Nevertheless, ground points were useful for the classification process due to the feature information these points provide. To test the robustness of the classifier, the evaluation metrics described in the previous chapter, Precision and Recall, were calculated for the datasets in study. From these metrics it was possible to calculate

the Overall Accuracy of the classifier. These metrics were obtained from the comparison of the manually labelled datasets and the classification results attained. The succeeding table, **Table 4.2** presents their percentiles.

Table 4.2: Detailed information of classification metrics, where synthetic data is represented by S# and real data is represented by R#.

| Sets | Vegetation | | Pylons | | Lines | | Total |
|------|------------|--------|-----------|--------|-----------|--------|------------------|
| | Precision | Recall | Precision | Recall | Precision | Recall | Overall Accuracy |
| S1 | 99.57 | 86.58 | 59.398 | 100 | 81.651 | 75.163 | 87.757 |
| R1 | 99.982 | 99.54 | 23.613 | 90.421 | 100 | 85.66 | 99.442 |
| R2 | 100 | 97.704 | 5.021 | 100 | 23.938 | 79.968 | 97.647 |
| R3 | 100 | 98.653 | 4.331 | 100 | 46.532 | 96.458 | 98.648 |

To provide a better understanding of the preceding results, a visual representation of one of the resulting datasets after the classifier labelling is now displayed, the sample Dataset R3, in **Figure 4.2**. For clarity, the different classes are represented by dissimilar colors.

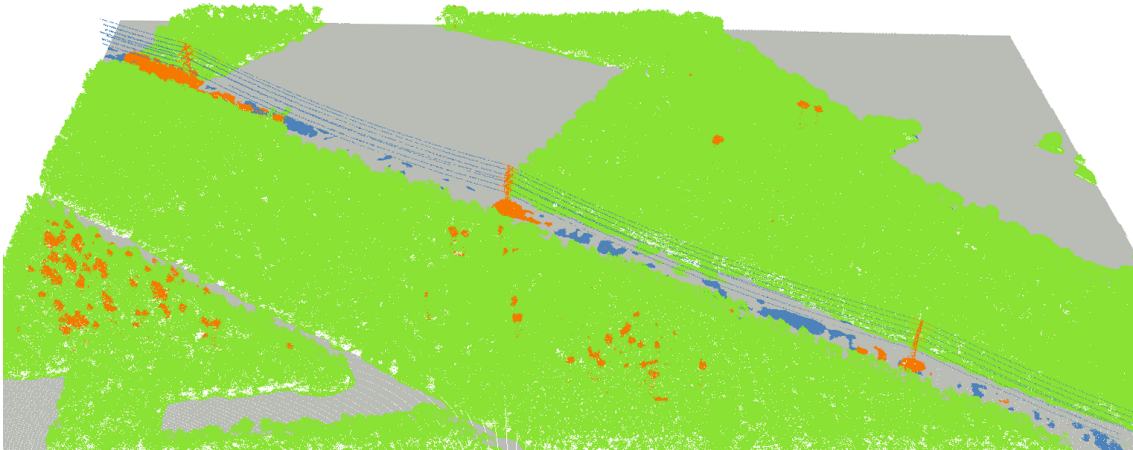
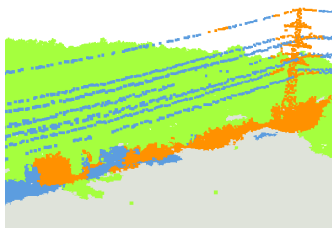


Figure 4.2: Resulting labelled data after Classification of the real life Dataset R3. Once more, Pylons are represented by the orange color, Lines are blue, Vegetation is green and Ground points are grey.

From the analysis of the previously provided figures, supported by the prior evaluation metrics, a few conclusions can be reached:

- While the Overall Accuracy values are generally good, it is important to note that it might not be an indication that all classes present good results. In other words, due to the overwhelming number of Vegetation points, when compared to Pylons and Lines, the Vegetation results will have a much bigger impact on the Overall Accuracy than the other two classes. Nevertheless, it is still a valuable metric to evaluate the classifier;

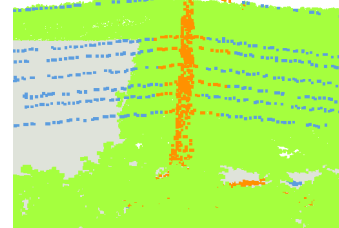
- In general, the number of relevant selected items by the classifier for each class label is positive, as observed from the Recall values, presenting nearly optimal values for Vegetation and Pylons. On the other hand, overall, the classifier shows some complications when selecting relevant Line points;
- Nevertheless, it is also possible to observe, from the Precision values, that being able to select relevant items is not intrinsically related to all the selected items being relevant. To put it another way, even though all the items from a specific class label can be categorised with the correct class label (thus achieving high Recall values), it does not necessarily mean items from different class labels are not incorrectly classified as the specified label (thus resulting in low Precision values). This issue is specially noticeable in Pylon and Line classification;
- Interpreting the visual representation of the classification results, it is possible to discern the most troublesome obstacles for the classifier used. The aforementioned issues are exemplified in **Figure 4.3** and they are:
 - Low Vegetation (i.e. tall grass or bushes) being incorrectly classified as Pylons or Lines. The first case is often true for Vegetation close or even in contact with the bases of the Pylons. The second scenario generally happens with Vegetation directly beneath the Lines;
 - Tall Vegetation, with geometric characteristics similar to Pylons, being misclassified as Pylons. This issue can occur in Vegetation close or far from the Pylons;
 - Line spans, namely connected to Pylon insulators, being mistakenly classified as Pylons.



(a) Low Vegetation misclassified as Pylons or Lines.



(b) Tall Vegetation incorrectly classified as Pylons.



(c) Line spans mistakenly classified as Pylons.

Figure 4.3: Close-ups of the most prominent issues present in the real life Dataset R3 faced by the classifier using the previous color code.

Despite the reasonable results achieved by the classifier there are still some challenges to overcome, as previously mentioned. The next subsection will emphasise on the classification post-processing results, including the methods used and the respective improvements.

4.2 Optimisation

This section will focus on the resolution of the previously stated issues of the classification process. These issues can be summed up as: Ground points wrongly classified as either Vegetation, Lines or Pylons; Vegetation being incorrectly classified as either Lines and Pylons; Line spans misclassified as Pylons. The proposed solutions (or suggested solutions in situations where it was not possible to optimise results) for the previously mentioned issues will be thoroughly explained, followed by the results obtained.

The solution for the first issue described, Ground Points incorrectly classified as other classes was to utilise a Progressive Morphological Filter. As explained before, this filter performs the segmentation of ground and non-ground returns based on morphological operations like erosion and dilation. Despite Ground points not being relevant to achieve the goal of this work, thus not being considered for evaluation purposes, it is still important to effectively separate them from points belonging to other classes.

The second issue mentioned, Low Vegetation being classified as Lines or Pylons, will actually be split in two, respectively. To tackle the first situation, a Passthrough Filter is utilised. Assuming that all Line points will have a minimum elevation, taking into account important parameters like the sag of the lines, every point below that elevation threshold can be filtered as a non-Line point. The second case, however, can be a lot more challenging. The proposed solution for this issue is an Euclidean Clustering algorithm, which segments objects into different clusters according to the euclidean distance between points and their own neighborhood. If this distance falls between a certain threshold, the points are categorised as the same object. The problem is when points from dissimilar objects fall within this distance threshold. For example, Vegetation points who are too close or even in contact with the base of pylons.

The next issue disclosed, Tall Vegetation being classified as Pylons, can also be either solved with ease or be a hurdle to overcome. Similarly to the last issue debated, the approach taken was to use an Euclidean Clustering Algorithm. This algorithm is able to effectively separate Tall Vegetation from Pylons when the distance between them is bigger than the defined distance threshold. On the other hand, for Tall Vegetation in close proximity with the Pylons this algorithm can be inaccurate. It is important to mention that a situation like this could be seriously hazardous were it to happen in real life, so it does not happen very often, but it is a situation that should be addressed and taken into consideration.

The last issue presented, Line spans incorrectly classified as Pylons, despite being the least problematic of the mentioned issues, could still be necessary to address in certain situations. Nevertheless, this issue can be hard to tackle when these Line spans are in close proximity or even in contact with Pylons or Pylon insulators. Even though it was not tested, one suggested solution is to use a catenary curve fitting algorithm in an attempt to isolate Line points from Pylon points.

In summary, the optimisation methods proposed are essentially filtering and segmentation algorithms. In particular, a Progressive Morphological Filter and Euclidean Clustering algorithms. Once more, the same evaluation metrics (Precision, Recall and Overall Accuracy) are utilised, in order to be able to compare the post-processing results with the prior classification results. The following table, **Table 4.3**, provides the percentiles of these metrics.

Table 4.3: Detailed information of optimisation metrics, where synthetic data is represented by S# and real data is represented by R#.

| Sets | Vegetation | | Pylons | | Lines | | Total |
|------|------------|--------|-----------|--------|-----------|--------|------------------|
| | Precision | Recall | Precision | Recall | Precision | Recall | Overall Accuracy |
| S1 | 100 | 92.625 | 66.924 | 100 | 100 | 75.163 | 92.385 |
| R1 | 100 | 99.764 | 36.959 | 100 | 100 | 86.088 | 99.685 |
| R2 | 100 | 99.89 | 32.7 | 100 | 100 | 83.502 | 99.835 |
| R3 | 100 | 99.976 | 61.272 | 100 | 100 | 97.011 | 99.969 |

For a better comprehension of the results now obtained, a visual representation of one of the attained datasets following the optimisation process is shown, once more the real life Dataset R3, in **Figure 4.4**. To make observations clearer dissimilar object classes are displayed in different sub figures, followed by a composition with all object classes for a general overview of the working environment after post-processing (using the same color scheme as before).



(a) Resulting Ground points after optimisation



(b) Resulting Line points after optimisation

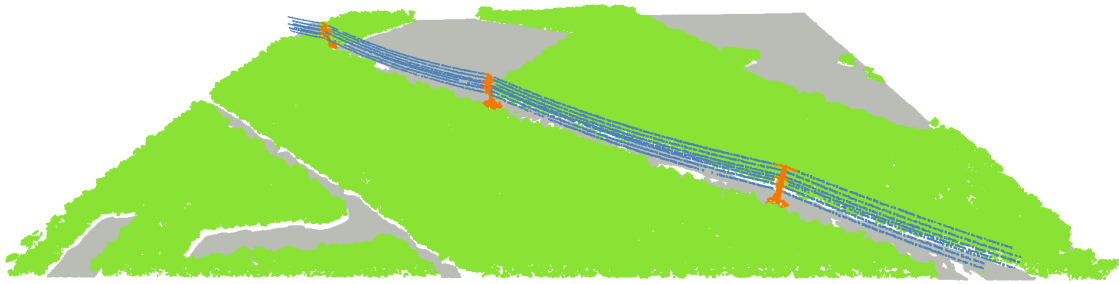


(c) Resulting Pylon points after optimisation

Figure 4.4: Optimisation results of the real life Dataset R3. Yet again, Pylons are represented by the orange color, Lines are blue, Vegetation is green and Ground points are grey.



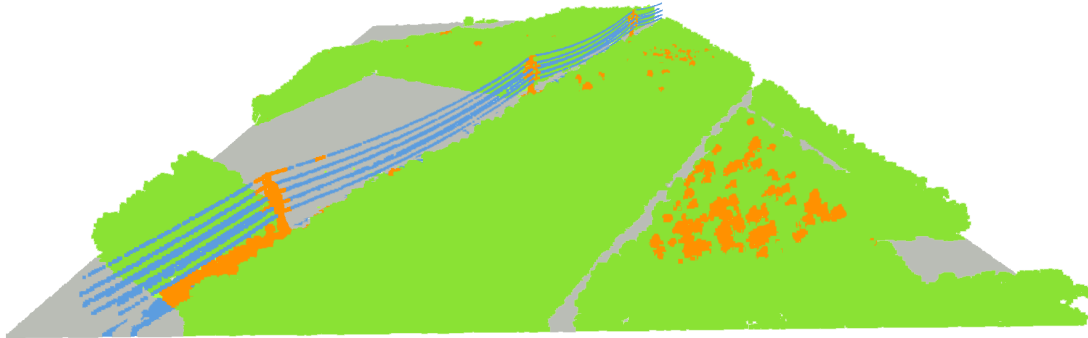
(d) Resulting Vegetation Points after optimisation



(e) General overview of the environment after optimisation

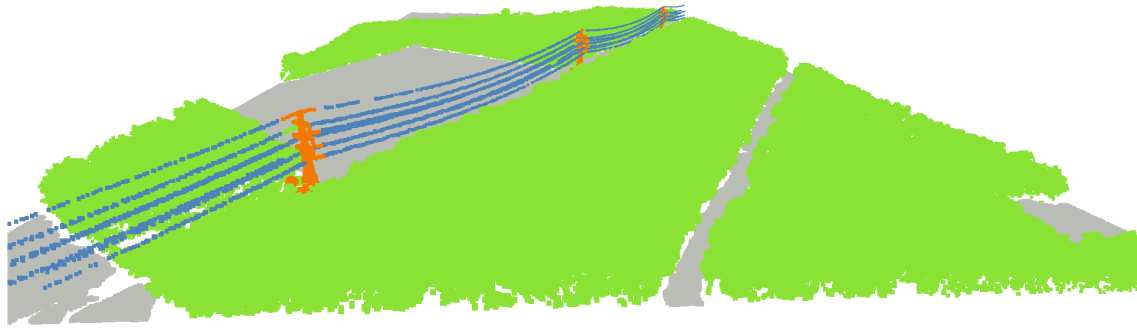
Figure 4.4: Optimisation results of the real life Dataset R3 (cont.).

To further illustrate the improvements attained with the optimisation process a graphical comparison of results is provided in **Figure 4.5**. This comparison presents a general overview of the real life Dataset R3 before and after the optimisation process.



(a) Classification results of the real life Dataset R3.

Figure 4.5: Comparison of Classification and Optimisation results of the real life Dataset R3 utilising the aforementioned color code. Pylons are orange, Lines are blue, Vegetation is green and Ground is grey.



(b) Optimisation results of the real life Dataset R3.

Figure 4.5: Comparison of Classification and Optimisation results of the real life Dataset R3 (cont.).

From the observation of the figure displayed before and the evaluation metrics of the respective optimisation results, in agreeance with the former classification results, a few conclusions can be made:

- Generally speaking, the usage of a Progressive Morphological Filter to isolate Ground and non-Ground points yielded positive results, as can be observed in the previous figure. Nevertheless, it is important to refer that, besides a few irregularities, the terrains in observation are relatively flat. For some rugged environments, it might be necessary to first normalise point cloud data. On the other hand, this might not always be a trivial task for large environments;
- Vegetation reaches notable values of Precision and Recall, meaning in general the post-processing algorithms were able to separate Vegetation points from points belonging to other object classes.

Passthrough filters proved effective to separate Low Vegetation points misclassified as Lines, under the assumption that Lines, even taking into account sag, should always have a higher elevation than Low Vegetation.

The same logic cannot be applied to Low Vegetation wrongly classified as Pylons. From the observation of the preceding figure it is possible to notice that Low Vegetation which is close or in contact with the bases of the Pylons are still a concern. This issue could be more or less aggravating, depending on the Vegetation species. To put it another way, whether it really is just Low Vegetation like bushes or small grass which will not grow much further, or if it is indeed a growing tree that could possibly endanger the integrity of the electrical structure. However, Low Vegetation farther from the Pylons was, in general, correctly optimised.

Furthermore, Tall Vegetation incorrectly classified as Lines, despite not happening very often, is still a concern. While the Euclidean Clustering algorithm works reasonably well when the distance between Lines and Vegetation is sizeable enough,

there is still not a coherent method to fix this issue when the distance threshold is surpassed.

Similarly, isolating Pylons from Tall Vegetation presents the same issue as separating the latter from Lines. However, this issue is more concerning since it happens more often due to certain Vegetation species having similar geometric characteristics as Pylons. Nevertheless, once again, when the distance threshold is not breached, the Euclidean Clustering algorithm is a reliable solution.

- An important observation about the Pylon optimisation results, which is intrinsically related to the issues that have been mentioned in the last topic, is that there is still a lot of room for improvement. Despite the notable Recall values, meaning relevant Pylon points are successfully extracted, the Precision values are mediocre which means there are still many Vegetation and Line points incorrectly classified as Pylons.
- On the other hand, Line points present good Precision values, meaning the post-processing methods to isolate Line points are effective, but the Recall values are only decent which is to say there are still relevant Line points which are being considered as, mostly, Pylons.

Notwithstanding the significant improvements achieved after post-processing the classified results, as mentioned during the previous result analysis, there are still more improvements to be made and there are still some issues to confront. Further details are provided in the following chapter, which will address the future work to be done, in order to complement this work, and the most important conclusions during the development of this work.

CONCLUSIONS AND FUTURE WORK

In this chapter, the conclusions attained during the development of this work are introduced in **section 5.1**, followed by the exposition of some guidelines for potential future researches in **section 5.2**.

5.1 Conclusions

Vegetation Management and Power Line Mapping using remote sensing techniques have been under scrutiny in many studies in recent years. The major reason behind this is: power line inspections must be regularly realised to assure a continuous distribution of electricity and most of the time this process is still manually done by field inspections, which can be very time consuming and labor-intensive.

From the previously presented state of the art analysis, based on different studies using different remote sensing techniques, a few conclusions have been reached:

- The described remote sensing techniques are situational. Some techniques can better monitor vegetation due to their ability to acquire precise information of the lower canopy layers, even in dense forest areas, while others are more suited for power line mapping, namely high resolution imagery which can provide valuable information about electric components. There are still remote sensing techniques, which, while not particularly good for vegetation management and power line monitoring, can be useful to assess damaged electric components;
- Similar to remote sensing techniques, the processing algorithms are also situational. While certain algorithms can present high accuracies for some specific tree species, they might not be as accurate as other algorithms for different tree species. Dense forest areas might also require different algorithms than rural areas or areas with

low density of vegetation. As such, in order to achieve accurate results, it is important to carefully select the most appropriate algorithm for the area over scrutiny;

- Despite the improvements of novel remote sensing techniques and algorithms, data processing is still essentially a manual procedure. Attempts to automate this process have been made. However, as explained in the previous points, each study area needs to be carefully analysed separately;
- Some remote sensing techniques have not been thoroughly studied for vegetation management and/or power line monitoring. One example being UAV-based studies, which have focused mostly on the specifications of UAV platforms instead of the methodology used. Existing studies have followed methods similar to ALS. Nevertheless, the applicability of UAV platforms for the referred purposes has been proven in literature. Furthermore, the number of studies in this area has increased in the latest years.

In this work, a UAV platform equipped with a LiDAR and the support of computer vision algorithms which were utilised to acquire and process point cloud data. Data was first submitted to a feature-based classifier in order to cluster different objects present in the working environment into their respective class labels (vegetation, pylons and lines). These results were later evaluated based on their Precision, Recall and Overall Accuracy. Some positive remarks can be attained from this classification process:

- Generally speaking the Recall values are acceptable, the lowest value being approximately 75% for Line points within the synthetic Dataset. Nevertheless, these values can be very positive for Vegetation and Pylon points, nearing 100% for the latter in most environments. What these values mean, in practice, is that the classifier can accurately retrieve relevant items from a specific class label out of all the items present in the environment;
- On the other hand, Precision values can be very unstable ranging from as low as 4% for Pylon points in one of the real life environments to nearly 100% for Vegetation and Lines in certain environments. What can be concluded with these results is that the classifier oftenly mistakes Vegetation or Line points for Pylon points;
- Despite the aforementioned issues, the Overall Accuracy of the classifier is still good, the lowest value being close to 87% for the synthetic Dataset and the highest value around 99% for the real life environments. The reason behind these values is the overwhelming number of Vegetation points which have very high Precision values when compared to other object classes.

Despite the overall positive results obtained with the classifier, there were still improvements to be made, namely to optimise Pylon results. The next step was to utilise segmentation and filtering techniques to improve said results. From this optimisation process a few conclusions can also be made:

- Recall values had few improvements, seeing that their values were already acceptable, one of them suffered no changes even. The reason behind this are some Line points being too close to Pylons sometimes making them hard to discern from Pylon insulators, for example;
- Precision values were severally improved, specially Line points which now round 100% for every environment. Pylon points were also refined but there's still a lot of room for improvement, since the values still range from about 33% to 67%. Nevertheless, the vast majority of the incorrect points are low Vegetation points in close proximity, or even in contact, with Pylons;
- Overall Accuracy values saw small improvements, since these values were already considerably high, the lowest value being nearly 92% for the synthetic Dataset while the values of the real life environments all round 99%.

Nevertheless, remote sensing techniques still present a few limitations before they can completely replace the traditional methods. During the development, of this work, the following challenges were met:

- Working with data collected from a real life environment does not always provide as good results as synthetic data, since simulated data allows a complete scrutiny over what data to collect and how to collect it, which might not always be possible in real life;
- Intrinsically related with the previous topic, LiDAR data in a real life environment will always present issues, like noise artifacts, the presence of outliers, sensor occlusion - to name a few. These issues might be more or less aggravating, based on the characteristics of the sensor, but they will always persist, which means, the classification models and the post-processing algorithms will have to be adaptive to overcome these issues, they cannot rely on the quality of the data available;
- Furthermore, despite the increase of available LiDAR data in recent years, the number of open LiDAR data repositories is still limited. In addition, finding an adequate dataset with specific characteristics for certain studies can be even more challenging;
- In spite of the apparent advantages the proposed method provides, in comparison with traditional methods, it still requires some work from a human operator, namely to generate the training sets of the classifier object and to tune important parameters of the post-processing algorithms;
- In general, however, the classification results were positive which means the methodology utilised is validated, even though these results are highly dependant on the quality of the training set. Moreover, the post-processing techniques that were used

were able to further improve these results to some extent, meaning these techniques are viable to overcome the respective issues described before. Nevertheless, there is still room for improvement as there are still concerning issues, namely when objects are in close proximity, or even connected, to each other.

5.2 Future Work

Despite the work done in this dissertation, there is still a lot of future work to do, namely further investigation which includes:

- Supplementary literary research is still required, particularly UAV-based studies, since some novel promising algorithms can be found, while some others can potentially be further validated;
- To complement the additional literary research, it is a good practice to benchmark the performance of the methods proposed in literature against a standard which is in accordance with the objective of the work in development.

Albeit the reasonable results attained with the proposed solution, some improvements can be made and additional work is needed to reach greater objectives.

- To further validate the proposed solution, and to get a better understanding of its limitations, it is necessary to test it using other, dissimilar, datasets. In particular, real datasets retrieved in a real-life environment;
- As seen previously, the used features are relevant and sufficient for the sample datasets tested in this work. However, other datasets could have different characteristics from the ones that were used. For example, certain datasets could present vegetation elements that have geometric shapes and measurements resembling pylons. For this reason, it is important to try different relevant features which could improve the classification results and possibly reduce the feature and training sets of the classifier;
- Furthermore, testing other classifiers, either supervised (like the Random Forest classifier) or unsupervised, could yield different results. In certain situations, the results can even be worse but have other advantages. For example, not requiring a human operator to define the feature set or create the training set;
- Similarly to the previous topic, testing different classification functions can also change the outcome of the classification process. Generally speaking global regularisation provides better results, but it is more computationally intensive in comparison to the other functions that were mentioned;

- Intrinsically related to the last points extracting more evaluation metrics can be extremely beneficial, not only to evaluate the classification results but also to assess the performance of certain method in comparison to a different method;
- For certain environments, specially when the terrain is rugged and has a wide range of elevation, it is more efficient to first normalise the input data and to generate a DTM, thus separating ground and non-ground points more accurately. However, it is important to take into account the computational resources expended during this process;
- In general, the post-processing results were positive but there is still a lot to improve. As previously mentioned, some Vegetation points in close proximity to Pylons are still wrongly classified. Further algorithm testing is required to find a more accurate solution. In addition, certain Line spans being mistakenly classified as Pylons is also another issue yet to overcome. One possible solution is to use line/catenary curve fitting models, but further investigation and testing is required. Following these corrections, it is possible to reconstruct electric structures, namely power lines;
- Taking into account all the previous improvements it is then possible to export the obtained results to an image processing software and to delineate fuel management bands, according to the respective legislation. Moreover, vegetation that could possibly endanger or compromise the electrical network can then be identified and measures can be taken by the respective entity who is responsible for electrical structure maintenance.

BIBLIOGRAPHY

- [1] *AirSim*. Accessed: 2020-09-29. URL: <https://microsoft.github.io/AirSim/>.
- [2] M. Awrangjeb. "Extraction of Power Line Pylons and Wires Using Airborne LiDAR Data at Different Height Levels." In: *Remote Sensing* 11 (July 2019), p. 1798. DOI: [10.3390/rs11151798](https://doi.org/10.3390/rs11151798).
- [3] S. A. Bello, S. Yu, and C. Wang. "Review: deep learning on 3D point clouds." In: *Remote Sensing* 12 (2020). DOI: [10.3390/rs12111729](https://doi.org/10.3390/rs12111729).
- [4] Y. Bengio, A. Courville, and P. Vincent. *Representation Learning: A Review and New Perspectives*. 2014. arXiv: [1206.5538 \[cs.LG\]](https://arxiv.org/abs/1206.5538).
- [5] J. L. Bentley. "Multidimensional binary search trees used for associative searching." In: *Communications of the ACM* 18 (Sept. 1975). DOI: [10.1145/361002.361007](https://doi.org/10.1145/361002.361007).
- [6] M. Bernstein. *Random Forests*. Accessed: 2021-06-08. URL: <http://pages.cs.wisc.edu/~matthewb/pages/notes/pdf/ensembles/RandomForests.pdf>.
- [7] C. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.
- [8] C. Chen, B. Yang, S. Song, X. Peng, and R. Huang. "Automatic Clearance Anomaly Detection for Transmission Line Corridors Utilizing UAV-Borne LIDAR Data." In: *Remote Sensing* 10 (Apr. 2018), p. 613. DOI: [10.3390/rs10040613](https://doi.org/10.3390/rs10040613).
- [9] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. "MeshLab: an Open-Source Mesh Processing Tool." In: 2008, pp. 129–136. DOI: [10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136](https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136).
- [10] J. Dandois and E. Ellis. "Remote Sensing of Vegetation Structure Using Computer Vision." In: *Remote Sensing* 2 (Aug. 2010). DOI: [10.3390/rs2041157](https://doi.org/10.3390/rs2041157).
- [11] *Dimensionality Reduction Algorithms: Strengths and Weaknesses*. Accessed: 2021-02-08. URL: <https://elitedatascience.com/dimensionality-reduction-algorithms>.
- [12] N. Donges. *A Complete Guide to the Random Forest Algorithm*. Accessed: 2021-03-15. 2019. URL: <https://builtin.com/data-science/random-forest-algorithm>.
- [13] S. Giraudot and F. Lafarge. "Classification." In: *CGAL User and Reference Manual*. 5.2. CGAL Editorial Board, 2020. URL: <https://doc.cgal.org/5.2/Manual/packages.html#PkgClassification>.

- [14] Y. Guo, H. Wang, H. Qingyong, H. Liu, L. Liu, and M. Bennamoun. “Deep Learning for 3D Point Clouds: A Survey.” In: *IEEE transactions on pattern analysis and machine intelligence* PP (2020). DOI: [10.1109/tpami.2020.3005434](https://doi.org/10.1109/tpami.2020.3005434).
- [15] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao, and L. Xiao. “A review of algorithms for filtering the 3D point cloud.” In: *Signal Processing: Image Communication* 57 (2017), pp. 103–112. DOI: [10.1016/j.image.2017.05.009](https://doi.org/10.1016/j.image.2017.05.009).
- [16] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen. “MeshCNN: A Network with an Edge.” In: *CoRR* abs/1809.05910 (2018). DOI: [10.1145/3306346.3322959](https://doi.org/10.1145/3306346.3322959).
- [17] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and B. Wolfram. “OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees.” In: *Autonomous Robots* (2013). Software available at <http://octomap.github.com>. DOI: [10.1007/s10514-012-9321-0](https://doi.org/10.1007/s10514-012-9321-0). URL: <http://octomap.github.com>.
- [18] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*. Springer, 2013.
- [19] J. Jensen and A. Mathews. “Assessment of Image-Based Point Cloud Products to Generate a Bare Earth Surface and Estimate Canopy Heights in a Woodland Ecosystem.” In: *Remote Sensing* 8 (Jan. 2016), p. 50. DOI: [10.3390/rs8010050](https://doi.org/10.3390/rs8010050).
- [20] P. Joshi. *What is Bootstrap Sampling in Statistics and Machine Learning?* Accessed: 2021-06-09. URL: <https://www.analyticsvidhya.com/blog/2020/02/what-is-bootstrap-sampling-in-statistics-and-machine-learning/>.
- [21] Y. Kobayashi, G. Karady, G. Heydt, and R. Olsen. “The Utilization of Satellite Images to Identify Trees Endangering Transmission Lines.” In: *Power Delivery, IEEE Transactions on* 24 (Aug. 2009), pp. 1703–1709. DOI: [10.1109/TPWRD.2009.2022664](https://doi.org/10.1109/TPWRD.2009.2022664).
- [22] S. Krig. “Ground Truth Data, Content, Metrics, and Analysis.” In: *Computer Vision Metrics: Survey, Taxonomy, and Analysis*. Berkeley, CA: Apress, 2014, pp. 283–311. ISBN: 978-1-4302-5930-5. DOI: [10.1007/978-1-4302-5930-5_7](https://doi.org/10.1007/978-1-4302-5930-5_7). URL: https://doi.org/10.1007/978-1-4302-5930-5_7.
- [23] *Laser (LAS) File Format Exchange Activities*. Accessed: 2021-01-29. URL: <https://www.asprs.org/divisions-committees/lidar-division/laser-las-file-format-exchange-activities/>.
- [24] M. Leeuwen, N. Coops, and M. Wulder. “Canopy surface reconstruction from a LiDAR point cloud using Hough transform.” In: *Remote Sensing Letters* 1 (Mar. 2010), pp. 125–132. DOI: [10.1080/01431161003649339](https://doi.org/10.1080/01431161003649339).
- [25] M. Lehtomäki, A. Kukko, L. Matikainen, J. Hyyppä, H. Kaartinen, and A. Jaakkola. “Power line mapping technique using all-terrain mobile laser scanning.” In: *Automation in Construction* 105 (Sept. 2019). DOI: [10.1016/j.autcon.2019.03.023](https://doi.org/10.1016/j.autcon.2019.03.023).

-
- [26] W. Li, Q. Guo, M. Jakubowski, and M. Kelly. "A New Method for Segmenting Individual Trees from the Lidar Point Cloud." In: *Photogrammetric Engineering and Remote Sensing* 78 (Jan. 2012), pp. 75–84. DOI: [10.14358/PERS.78.1.75](https://doi.org/10.14358/PERS.78.1.75).
- [27] A. S. Malik, T.-S. Choi, and H. Nisar. "Depth Map and 3D Imaging Applications: Algorithms and Technologies." In: 2011. DOI: [10.4018/978-1-61350-326-3](https://doi.org/10.4018/978-1-61350-326-3).
- [28] L. Matikainen, M. Lehtomäki, E. Ahokas, J. Hyyppä, M. Karjalainen, A. Jaakkola, A. Kukko, and T. Heinonen. "Remote sensing methods for power line corridor surveys." In: *ISPRS Journal of Photogrammetry and Remote Sensing* 119 (Sept. 2016), pp. 10–31. DOI: [10.1016/j.isprsjprs.2016.04.011](https://doi.org/10.1016/j.isprsjprs.2016.04.011).
- [29] D. Maturana and S. Scherer. "VoxNet: A 3D Convolutional Neural Network for real-time object recognition." In: 2015, pp. 922–928. DOI: [10.1109/IR0S.2015.7353481](https://doi.org/10.1109/IR0S.2015.7353481).
- [30] D. Meagher. *Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer*. Tech. rep. IPL-TR-80-111. Rensselaer Polytechnic Institute, Image Processing Laboratory, Oct. 1980.
- [31] S. Mills, M. P. Gerardo-Castro, Z. Li, J. Cai, R. Hayward, L. Mejias, and R. Walker. "Evaluation of Aerial Remote Sensing Techniques for Vegetation Management in Power-Line Corridors." In: *IEEE Transactions on Geoscience and Remote Sensing* 48 (Oct. 2010). DOI: [10.1109/TGRS.2010.2046905](https://doi.org/10.1109/TGRS.2010.2046905).
- [32] N. Munir, M. Awrangjeb, B. Stantic, G. Lu, and S. Islam. "Voxel-based extraction of individual pylons and wires from lidar point cloud data." In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-4/W8 (Sept. 2019), pp. 91–98. DOI: [10.5194/isprs-annals-IV-4-W8-91-2019](https://doi.org/10.5194/isprs-annals-IV-4-W8-91-2019).
- [33] A. Nguyen and B. Le. "3D point cloud segmentation: A survey." In: Nov. 2013, pp. 225–230. ISBN: 978-1-4799-1201-8. DOI: [10.1109/RAM.2013.6758588](https://doi.org/10.1109/RAM.2013.6758588).
- [34] P. Nygren and M. Jasinski. "A Comparative Study of Segmentation and Classification Methods for 3D Point Clouds." Master's thesis. Gothenburg, Sweden: Chalmers University of Technology, 2016.
- [35] *Open.NRW – Das Open-Government-Portal in NRW*. Accessed: 2021-01-27. URL: <https://open.nrw/>.
- [36] *Portugal Wildfires*. Accessed: 2020-01-31. URL: <http://www.portugalwildfires.com/portugal-wildfires-latest-news/>.
- [37] T. C. Project. *CGAL User and Reference Manual*. 5.2. CGAL Editorial Board, 2020. URL: <https://doc.cgal.org/5.2/Manual/packages.html>.
- [38] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation." In: 2017, pp. 77–85. DOI: [10.1109/CVPR.2017.16](https://doi.org/10.1109/CVPR.2017.16).

- [39] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. "ROS: an open-source Robot Operating System." In: *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*. Kobe, Japan, May 2009.
- [40] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)." In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011.
- [41] M. Rutzinger, A. Pratihast, S. Oude Elberink, and G. Vosselman. "Detection and Modelling of 3D Trees from Mobile Laser Scanning Data." In: vol. 38. Jan. 2010.
- [42] S. Tang, P. Dong, and B. Buckles. "Three-dimensional surface reconstruction of tree canopy from lidar point clouds using a region-based level set method." In: *International Journal of Remote Sensing* 34 (Feb. 2013), pp. 1373–1385. DOI: [10.1080/01431161.2012.720046](https://doi.org/10.1080/01431161.2012.720046).
- [43] G. Turk. *The PLY Polygon File Format*. Accessed: 2021-01-29. 1994. URL: <https://web.archive.org/web/20161204152348/http://www.dcs.ed.ac.uk/teaching/cs4/www/graphics/Web/ply.html>.
- [44] *Velodyne LiDAR PUCK™*. Velodyne Acoustics, Inc. 2015.
- [45] N. Verma, D. Lamb, N. Reid, and B. Wilson. "Comparison of Canopy Volume Measurements of Scattered Eucalypt Farm Trees Derived from High Spatial Resolution Imagery and LiDAR." In: *Remote Sensing* 8 (May 2016), p. 388. DOI: [10.3390/rs8050388](https://doi.org/10.3390/rs8050388).
- [46] W. Wagner, M. Hollaus, C. Briese, and V. Ducic. "3D vegetation mapping using small-footprint full-waveform airborne laser scanners." In: *International Journal of Remote Sensing* 29 (Mar. 2008), pp. 1433–1452. DOI: [10.1080/01431160701736398](https://doi.org/10.1080/01431160701736398).
- [47] S. Walk. *ETH Zurich Random Forest Template Library*. Accessed: 2021-03-24. 2014. URL: https://prs.igp.ethz.ch/research/Source_code_and_datasets.html.
- [48] L. Wallace, A. Lucieer, Z. Malenovský, D. Turner, and P. Vopenka. "Assessment of Forest Structure Using Two UAV Techniques: A Comparison of Airborne Laser Scanning and Structure from Motion (SfM) Point Clouds." In: *Forests* 7 (Mar. 2016), p. 62. DOI: [10.3390/f7030062](https://doi.org/10.3390/f7030062).
- [49] Y. Wang, Q. Chen, L. Liu, and K. Li. "A Hierarchical unsupervised method for power line classification from airborne LiDAR data." In: *International Journal of Digital Earth* (July 2018), pp. 1–17. DOI: [10.1080/17538947.2018.1503740](https://doi.org/10.1080/17538947.2018.1503740).
- [50] Y. Wang, H. Weinacker, and B. Koch. "A Lidar Point Cloud Based Procedure for Vertical Canopy Structure Analysis And 3D Single Tree Modelling in Forest." In: *Sensors* 8 (June 2008a), pp. 3938–3951. DOI: [10.3390/s8063938](https://doi.org/10.3390/s8063938).

- [51] Y. Wang, H. Weinacker, B. Koch, and K. Stereńczak. "Lidar point cloud based fully automatic 3D single tree modelling in forest and evaluations of the procedure." In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives* 8 (June 2008b), pp. 1682–1750.
- [52] D. Wanik, J. Parent, E. Anagnostou, and B. Hartman. "Using vegetation management and LiDAR-derived tree height data to improve outage predictions for electric utilities." In: *Electric Power Systems Research* 146 (May 2017), pp. 236–245. DOI: [10.1016/j.epsr.2017.01.039](https://doi.org/10.1016/j.epsr.2017.01.039).
- [53] J. White, M. Wulder, M. Vastaranta, N. Coops, D. Pitt, and M. Woods. "The Utility of Image-Based Point Clouds for Forest Inventory: A Comparison with Airborne Laser Scanning." In: *Photogrammetric Engineering and Remote Sensing* 4 (June 2013), pp. 518–536. DOI: [10.3390/f4030518](https://doi.org/10.3390/f4030518).
- [54] C. Zhang, Y. Zhou, and F. Qiu. "Individual Tree Segmentation from LiDAR Point Clouds for Urban Forest Inventory." In: *Remote Sensing* 7 (June 2015), pp. 7892–7913. DOI: [10.3390/rs70607892](https://doi.org/10.3390/rs70607892).
- [55] K. Zhang, S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, and C. Zhang. "A progressive morphological filter for removing nonground measurements from airborne LIDAR data." In: *IEEE Transactions on Geoscience and Remote Sensing* 41 (2003), pp. 872–882. DOI: [10.1109/TGRS.2003.810682](https://doi.org/10.1109/TGRS.2003.810682).
- [56] L. Zhu and J. Hyypä. "Fully-Automated Power Line Extraction from Airborne Laser Scanning Point Clouds in Forest Areas." In: *Remote Sensing* 6 (Nov. 2016), pp. 11267–11282. DOI: [10.3390/rs6111267](https://doi.org/10.3390/rs6111267).

