



Master Thesis

submitted within the UNIGIS MSc programme
Interfaculty Department of Geoinformatics - Z_GIS
University of Salzburg

Detection of canopy layers in forests using density based segmentation of point clouds

by

Karsten Kehlert

105487

A thesis submitted in partial fulfilment of the requirements of
the degree of
Master of Science – MSc

Advisor: Dr Ivan Tomljenovic

Place: Salzburg, Austria

Date: 01.01.2024

Science Pledge

I hereby declare that the thesis is entirely the result of my work. Artificial intelligence has been implemented as a linguistic tool to improve writing or suggest snippets of code in programming, but in no case has it produced a text or an entire code. I have cited all sources I have used in my thesis, I have always indicated their origin. This thesis was not previously presented to another examination board and has not been published.

Salzburg, 01.01.2024

Acknowledgments

I would like to express my sincere gratitude to Dr. Ivan Tomljenovic for his excellent and dedicated mentorship throughout my research on dead wood detection and the thesis. Special thanks go also to the awesome UNIGIS team: they did support me kindly during the entire course of study, thank you to each and every one of you. The Bavarian Forest National Park administration did provide the data - thank you for your cooperation and trust.

A big thank you also goes to the developers and communities behind the open source software solutions: for programming to R, for spatial visualizations to QGIS, for graphical post-processing to Inkscape, for office software to Open Office, for language services to DeepL Translator and ChatGPT. Without their constant effort and dedication, this thesis would not have been possible.

Abstract

The detection of trees in large forest areas using LiDAR data is limited by the high computational effort required to process raw point clouds. Nevertheless, capturing this information is crucial for effective forest management and conservation, especially in the context of monitoring the effects of climate change on forests, where detailed tree-level information is essential. To accomplish this task, profiler methods have been developed, among others, in which point cloud data is vertically structured and abstracted so that it can subsequently be processed as raster data. Although this approach significantly reduces the computational effort, it can lead to the loss of critical information required for accurate results. In this study, a profiler algorithm based on point cloud density is replicated and evaluated. The aim is to evaluate its effectiveness in estimating canopy layers. This is determined by comparing tree canopy metrics from forest inventory data and the corresponding metrics in the canopy model. The LiDAR dataset and the evaluation dataset for this study come from long-term monitoring sites in the Bavarian Forest National Park in Germany. Both sets were acquired in 2016. The LiDAR data has an average point density of 35 points per square meter, and the monitoring inventory data for the evaluation comprises about 2800 trees on 87 plots. The forest structure on these plots is diverse, with different proportions of broadleaf and coniferous trees along an ecological and climatic altitude gradient. The application of the replicated algorithm to the point cloud data generates a canopy model consisting of individual canopy layers. This model was evaluated on a selection of dominant and subdominant trees and yielded a modest overall fit with a mean canopy accuracy of 146 % and a mean canopy precision of 80 %. Further modifications and testing on point clouds with higher point densities are required to adapt and effectively utilize this replicated algorithm for estimating canopy layers in forests.

Zusammenfassung

Die Erkennung von Bäumen in großen Waldgebieten mit Hilfe von LiDAR-Daten ist durch den hohen Rechenaufwand begrenzt, der für die Verarbeitung der rohen Punktwolken erforderlich ist. Dennoch ist die Erfassung dieser Informationen für eine effektive Waldbewirtschaftung und -erhaltung von entscheidender Bedeutung, insbesondere im Zuge der Anpassung des Waldes an den Klimawandel, in der ein detailliertes Monitoring auf Baumebene unerlässlich ist. Zur Bewältigung dieser Aufgabe wurden unter anderem Profilermethoden entwickelt, bei denen Punktwolkendaten vertikal strukturiert und abstrahiert werden, um sie dann als Rasterdaten verarbeiten zu können. Dieser Ansatz reduziert den Rechenaufwand erheblich, kann aber zu einem Verlust von wichtiger Information führen, die für das Erzielen von präzisen Ergebnissen notwendig ist. In dieser Studie wird ein auf der Punktwolkendichte basierender Profileralgorithmus repliziert und evaluiert. Ziel ist es, seine Wirksamkeit in der Schätzung von Kronenschichten zu bewerten. Diese wird anhand eines Vergleichs zwischen Baumkronenmetriken aus Waldinventurdaten und deren entsprechenden Metriken im Kronenmodell ermittelt. Der LiDAR-Datensatz und der Inventurdatensatz für diese Studie stammen von Langzeitüberwachungsstandorten im Nationalpark Bayerischer Wald in Deutschland. Beide Datensätze wurden im Jahr 2016 erhoben. Die LiDAR-Daten haben eine mittlere Punktdichte von 35 Punkten pro Quadratmeter, und die Inventurdaten umfassen etwa 2800 Bäume auf 87 Parzellen. Die Waldstruktur auf diesen Flächen ist vielfältig und weist unterschiedliche Anteile von Laub- und Nadelbäumen entlang eines ökologischen und klimatischen Höhengradienten auf. Wendet man den replizierten Algorithmus auf die Punktwolkendaten an, so wird ein Baumkronenmodell erstellt, das sich aus einzelnen Baumkronenschichten zusammensetzt. Dieses Modell wurde an einer Auswahl von dominanten und subdominanten Bäumen evaluiert. Es schneidet im Mittel über alle Parzellen mäßig ab: die mittlere Kronengenauigkeit beträgt 146 % (canopy accuracy) und die mittlere Kronenpräzision 80 % (canopy precision). Um diesen rekonstruierten Algorithmus für die Schätzung von Baumkronenschichten in Wäldern verwenden zu können, muss er modifiziert und an Punktwolken mit höherer Punktdichte getestet werden.

Content

Acknowledgement	II
Abstract	
English Version	III
German Version	IV
1. Introduction	
1.1 Forest Management Based On Tree-Level Information	1
1.2 The Collection of Tree-Level Information	1
1.3 Tree Detection with LiDAR	1
1.4 Point Cloud Segmentation	2
1.5 Research Objectives	3
1.6 Research Approach	3
1.7 Research Impact	3
2. Material and Methods	
2.1 Study Area	4
2.2 Data	5
2.3 Programming	6
2.3.1 Key Concepts	6
2.3.2 Software	7
2.3.3 Paradigms	7
2.3.4 Structure	8
2.4 The Replicated Algorithm	9
2.4.1 Stratification	10
2.4.1.1 The Stratification Module	10
2.4.1.2 Pre-Processing of the Input Data	10
2.4.1.3 Output Data Structure	11
2.4.1.4 Processing	11
2.4.1.5 Key Terminology	12

2.4.2	Evaluation	12
2.4.2.1	The Evaluation Module	13
2.4.2.2	Matching Inventory Metrics and Canopy Model	14
2.4.3	Feature Importance Analysis	14
2.4.3.1	Pattern Definition and Grouping of the Data	14
2.4.3.2	Computing the Feature Importance Analysis	14
3.	Results	15
4.	Discussion	
4.1	The Performance of the Replicated Algorithm	17
4.1.1	Canopy Accuracy and Canopy Precision	17
4.1.2	Ranking	18
4.2	The Internal Structure of the Replicated Algorithm	19
4.2.1	Tree Dominance	19
4.2.2	Resolution	19
4.2.3	Smoothing	20
4.2.4	Evaluation Kernel	21
5.	Outlook	21
6.	Conclusion	22
	References	23
	Appendix A - Processing Module	
A.1	Data and Data Structures	i
A.2	Iteration in the Stratification Module	i
A.3	Canopy Layer Detection	ii
A.4	Evaluation Module	iii
	Appendix B - Feature Importance	
B.1	Patterns and Indicators at Plot Level	iv
B.2	Patterns and Indicators at Tree Level	v

Appendix C - Canopy Models from Mode 1 and Mode 2

C.1 - C.2 Tree Point Cloud	vi
C.3 - C.5 Tree Surface Model	vi

Appendix D - Plot Point Clouds and CHMs

D.1 Open Land	vii
D.2 Woodland	vii
D.3 Forest with Gaps	viii
D.4 Closed Forest	viii
D.5 3D Canopy Surface Model	ix

Appendix E - Code Examples from the Stratification Module

E.1 Canopy Layer Detection	x
E.2 Cylindrical Stratification Kernel	xi
E.3 Smoothing	xii
E.4 Canopy Layer Boundaries	xiii

Table of Figures

Figure 2.1.1	The National Park area within Germany	4
Figure 2.1.2	Bavarian Forest National Park region with transects	4
Figure 2.2	Height profile of transect 4	5
Figure 2.3.1	Functional programming and packages	8
Figure 2.3.2	Modular programming at project level	9
Figure 2.4.1	Analysis and processing ranges around a plot	10
Figure 2.4.2	Analysis raster grid	11
Figure 2.4.3	Canopy accuracy and canopy precision	13
Figure 3.1	Ranking of patterns in the feature importance analysis	16
Figure A.1	Data and data structures	i
Figure A.2	Iteration in the stratification module	i
Figure A.3	Canopy layer detection	ii
Figure A.4	Detection of the best fit in the canopy model	iii
Figure C.1	Tree point cloud, top view	vi
Figure C.2	Tree point cloud, side view	vi
Figure C.3	Canopy layer 2, upper boundary, Mode 1	vi
Figure C.4	Canopy layer 2, upper boundary, Mode 2	vi
Figure C.5	Canopy layer 3, upper boundary, Mode 1	vi
Figure D.1	Point cloud and CHM: open land	vii
Figure D.2	Point cloud and CHM: woodland	vii
Figure D.3	Point cloud and CHM: forest with gaps	viii
Figure D.4	Point cloud and CHM: closed forest	viii
Figure D.5	3d Canopy Surface Model	ix

Index of tables

Table 3.1	Canopy model evaluation	15
Table 3.2	Pattern index	16
Table B.1	Patterns and indicators at plot level	iv
Table B.2	Patterns and indicators at tree level	v
Table E.1	Code example: canopy layer detection	x
Table E.2	Code example: cylindrical stratification kernel	xi
Table E.3	Code example: smoothing	xii
Table E.4	Code example: canopy layer boundaries	xiii

Acronyms

AI	artificial intelligence
ALS	Aerial Laser Scanner
a.s.l.	above sea level
BIOKLIM	Biodiversity and Climate Change Project
CHM	Canopy Height Model
DBH	diameter at breast height
GNSS	Global Navigation Satellite System
IQR	interquartile range
LiDAR	Light Detection and Ranging
SD	standard deviation

1. Introduction

1.1 - Forest Management Based on Tree-Level Information

The effective adaptation of forest conservation and management strategies to the challenges of climate change depends on accurate and detailed information on the health and condition of forests. The more diverse and heterogeneous a forest is, the less accurate is a generalized form of information. Therefore, segmenting a forest into homogenous sub-units such as forest stands is a practical solution for conventionally managed forests. However, it is still a generalized level of information that may or may not be applicable to different forest types.

At a time when it is possible to monitor, model and evaluate the health and condition of individual trees on a large scale, forest conservation and management strategies can be adapted based on tree-level information. For example, the sensitivity or resilience of individual trees or tree species to particular climatic conditions can be identified and taken into account in future forest management decisions.

1.2 - The Collection of Tree-Level Information

In forest management, there are two approaches to collecting information at tree-level: traditional forest inventories and Remote Sensing techniques. In traditional forest inventories, precise information about individual trees is collected on site and extrapolated to the stand and forest level. These inventories are expensive [1], time-consuming, and therefore limited to small sample areas. Remote Sensing can be used to collect tree-level information over a large area, but its accuracy is limited and therefore still subject to research [2]. Once this accuracy issue of Remote Sensing techniques is solved, forest health monitoring can be automated through area-wide inventories at tree-level with high temporal and spatial resolution [2,3].

The key issue with the Remote Sensing approach is the accurate identification of individual trees in a forest. Forests can have different layers of vegetation, and trees can be present in each layer. The detection of trees therefore requires a three-dimensional analysis that includes both horizontal and vertical delineations. When using aerial or spaceborn sensor data, the most visually prominent part of a living tree is its canopy. Accordingly, a tree detection with this data is based on the delineation of tree canopies.

1.3 - Tree Detection with LiDAR

Within Remote Sensing technologies, Light Detection and Ranging (LiDAR) technology enables accurate measurement of tree canopy architectures [4,5]. As it is an active

sensor, LiDAR is insensitive to interference from illumination and shadows [6]. Advances in LiDAR technology in recent years have resulted in data with high spatial resolution and point density. These advances have not only refined canopy measurements, but have also enabled accurate detection of trees in the understory [7].

Consequently, the methodology for segmenting trees in point clouds from Aerial Laser Scanner (ALS) is based on the detection of whole tree canopies or tree tops. Traditionally, there are two main approaches for this segmentation task: geometric methods that work with the point cloud and analyze the distribution of points in space [8,9]. And raster-based methods that analyze the information from the point cloud stored in a raster surface model, such as the watershed algorithm [10,11].

To compare the performance and results of the methods of both approaches, they were tested in benchmark studies under the same conditions to determine which approach and method is best suited for specific environments and tasks [12,13,14,15].

The results of all methods are comparable in temperate and cold climate forests. They depend more on the forest structure than on the algorithm used [13], and both approaches have difficulties in multi-story stands with complex forest structure [14]. Compared to the extreme density and structural complexity of tropical rainforest, geometric methods perform better in segmenting small to medium sized vegetation, while raster based methods are better for medium to large vegetation [15].

There is a major trade-off between the two approaches: raster methods are computationally less demanding than geometric methods and can therefore be easily applied to large areas. However, information is lost, especially the 3D information of a point cloud. In the case of a Canopy Height Model (CHM), the analysis is limited to the overstory canopies [8]. Geometric methods detect both the over- and the understory canopies as 3D objects [9], but the application to large areas is limited due to the high computational effort.

1.4 - Point Cloud Segmentation

These results are all based on the input of raw point clouds where the relationship between the points is not defined. The authors in [16] found that structuring a point cloud benefits tree segmentation. One way to structure a point cloud is to divide the vertical profile of the forest point cloud into horizontal layers. This can be done mathematically by grouping percentiles of a point cloud [17], or by using meaningful units such as canopy layers.

The derivation of canopy layers from point density is an approach to forest stratification in ALS point clouds. According to this approach, regions with high point density are defined as tree canopies [7,15,17,18]. The reference space for measuring density can be 2D or 3D: The authors in [19] use 2D horizontal profiles, in [20] voxels with fixed size, and in [7] a cylinder with flexible size that adapts to local conditions.

What all of these approaches have in common is that they provide the 3D information from the point cloud in raster surface layers, enabling volumetric analysis based on raster layers. This strategy offers the potential to shift the computationally intensive point cloud analysis into raster space, enabling the scalability of tree segmentation for large-scale applications.

1.5 - Research Objectives

This study aims to replicate the canopy stratification method presented by Hamraz et al. [7]. It further evaluates the fit between the resulting canopy model and tree metrics from inventory data. Finally, the success of the replication is evaluated and recommendations for its further development are given.

1.6 - Research Approach

The replication of the canopy stratification algorithm is achieved by functional programming. The programming environment is R. The processing workflow is semi-automated and based on scripts.

It is important to emphasize that this study is an interpretation of the method of Hamraz et al. and not a copy of the original approach. In the following, the term *replicated algorithm* is used to refer to this interpreted version.

1.7 - Research Impact

The stratification of a forest into canopy layers represents a substantial gain in the derivation of forest parameters and in the detection of individual trees. It also contributes to the automated monitoring of tree health on large scales.

2. Material and Methods

2.1 Study Area

The Bavarian Forest National Park (Figure 2.1.1) is located in the southeast of Germany (N 49° 0' 38.0 E 13° 22' 33.5). It was founded in 1970, and covers about 24000 ha (Figure 2.1.2) of a former commercial forest at altitudes between 650 and 1420 m a.s.l. The climate is temperate with Atlantic and continental influences. The average annual air temperature ranges from 9.7 °C in the lowlands to 3.4 °C in the high montane areas. The total annual precipitation ranges from 900 mm in the lowlands to 1800 mm in the high montane areas. The geologic substrate is homogeneous throughout the park area, landforms are rounded, and soils are acidic. [21,22]

In the National Park region there are two forms of land management: zones of wilding without anthropogenic interference, and zones of forestal management. The region is also characterized by an altitude gradient (Figure 2.2), that determines climatic and ecological characteristics. [21,22]

Over 80 % of the park is covered by forest. The dominant tree species are Norway spruce (*Picea abies*) and European beech (*Fagus sylvatica*). Other tree species include silver fir (*Abies alba*), common rowan (*Sorbus aucuparia*), sycamore maple (*Acer pseudoplatanus*), and birch (*Betula pendula*). According to the tree species composition given in [22], Norway spruce (85%) dominates the high montane area above 1100 m a.s.l. with some European beech (12 %). In the montane area between 900 and 1100 m a.s.l. European beech (50%) is more common than Norway spruce (30 %) and silver fir (10%). Due to frequent windthrow and bark beetle infestation, the forest is rich in structures and shows different stages of forest succession.

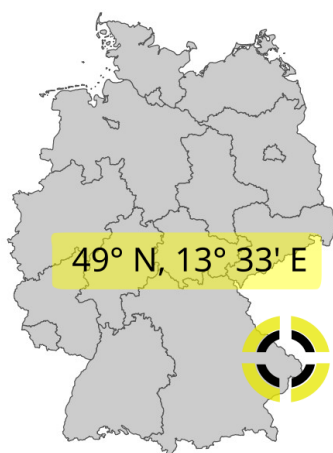


Figure 2.1.1: The National Park area within Germany

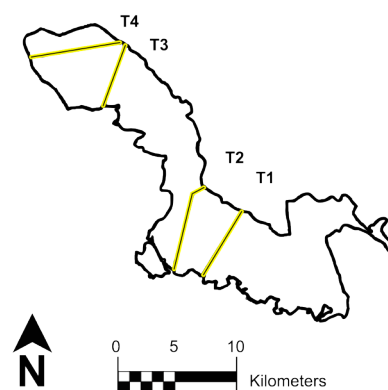


Figure 2.1.2: Bavarian Forest National Park region with transects

2.2 Data

As part of the Biodiversity and Climate Change Project (BIOKLIM), the Bavarian Forest National Park has established long-term experimental plots to monitor forest structure and biodiversity. These plots are located along 4 transects (Figure 2.1.2) covering both management types, the altitude gradient and the local diversity of the National Park region. According to the methodology described in [21], field surveys were conducted in 2006 and 2016.

The 2016 inventory is the reference dataset for evaluating the results of the canopy stratification. 87 plots were selected across 4 transects. Metrics, position and descriptions of individual trees with a diameter at breast height (DBH) of 7 cm or greater were recorded. The tree position was determined using a Leica GS 14 Professional GNSS receiver [21]. 75 % of all trees have a location accuracy of less than 55 cm. The tree height was measured with a Vertex III system [23].

The ALS point cloud data is from October 2016, and was captured during the transition from leaf-on to leaf-off conditions. The data set has an average point density of 35 points per square meter. The analysis uses only the XYZ coordinates from the point cloud. Attributes beyond these coordinates are not subject to the study. The data is stored in a v1.2 LAS file format and were generously provided by the administration of the Bavarian Forest National Park.

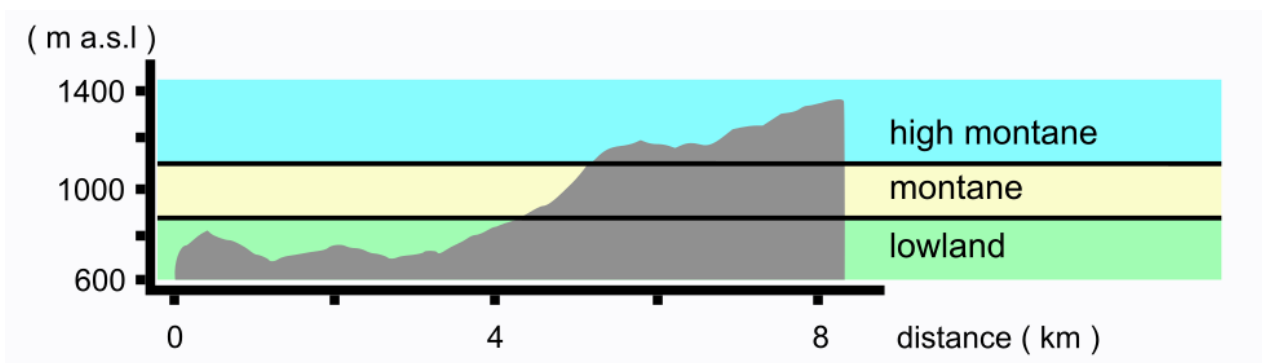


Figure 2.2: Height profile of transect 4, adapted from [21]

2.3 Programming

The task of replicating the algorithm of Hamraz et al. relies on programming. Guidelines for this programming can be found in [7,24]. Where details or descriptions are missing own solutions have been crafted, making the replication of the algorithm rather an interpretation than a copy.

This section describes the principles used to replicate the algorithm of Hamraz et al. For a comprehensive, detailed description, please refer to Section 2.4. Additionally, Appendix A provides a schematic representation of the processing modules, while Appendix E includes code examples from functions that are implemented in the stratification module.

2.3.1 - Key Concepts

There is a basic rule in programming: "Don't repeat code". The origin of this rule lies in efficiency: repeating code from other authors is a waste of time and effort. Repeating own code increases the time and effort required for updating and maintenance.

For example, an expression is copied into the code structure as plain text each time it is required. On the one hand, this increases the amount of code to be tested and maintained. On the other hand, if this expression is to be changed, it must be changed at every single point where it is used. This may not be an issue with a few lines of code, but with a complex and extensive code structure it becomes time-consuming and error-prone.

Programming therefore begins with an overview of reusable code structures and coding environments, and continues with the selection of those that can be used to build the algorithm. Specifically, this involves selecting software to help with coding, testing and debugging, as well as selecting related work such as packages that reduce the amount of code to be written.

When it comes to programming, the use of external resources is the key to increasing efficiency. In addition to software and software packages, there is the exchange with other programmers on digital platforms or in communities, as well as the use of artificial intelligence (AI) to automate the creation of code components. At this stage, publicly available applications that use AI, especially large language models, are not able to create code for complex tasks or create code that is compatible with recent updates to software or packages. Therefore, the activity of programmer communities, which discuss issues, keep software and packages up to date and document them, is still an important selection criterion for the software environment.

2.3.2 - Software

Choosing among open source solutions, there is one prominent package that addresses LiDAR point clouds in the context of forest management and forest research: it is the lidR package from J.R. Roussel [25]. It is based on R [26], implements C++ for efficient processing, covers all point cloud analyses needed for this study, it is up-to-date, actively maintained, and well documented. The R environment offers a lot more: for raster analysis there is the terra package [27], for visualizations of point clouds and raster surfaces there is the rgl package [28], and the R-Studio suite [29] provides convenient solutions for code development.

So for replicating the algorithm from Hamraz et al. the choice was R and the R environment. The package versions of the main packages used in this study were version 4.0.3 for the lidR package, the version 1.7-4.1 for the terra package and the version 1.2.1 for the rgl package.

2.3.3 - Paradigms

The next step in programming is the choice of methodology. There are different paradigms in programming, the one that best fits the needs of this study is functional programming. It meets the requirements of scientific research, offers transparency and repeatability, and embodies the key concept of programming "Don't repeat code". Functional programming is an approach to software development that focuses on functions.

Functions are discrete units that fulfill a specific task. They always provide the same output for a specific input and have no side effects. A function is defined once, its definition is kept in one place, and the function can be used by calling the function. A function call is a short expression consisting of the function name followed by the arguments. These arguments are the input data for a function. Since a function is a closed unit with its own environment, it attempts to act on every input object, but it cannot act on anything outside the function environment.

For example, a function is implemented to slice an input object. If the object is sliceable, the output of the function is a sliced object. If the object is not sliceable, the function throws an error. Only objects that are passed as arguments to the function can be sliced.

The result of a function is therefore predictable, all components are defined, and the entire process is transparent. And functions are modular like building blocks: they can be chained and combined to solve complex tasks, and they can take other functions as input. This ensures that each step and each intermediate result in the functional programming workflow can be accessed and checked individually.

2.3.4 - Structure

Writing a function for each specific task fills up the code and its environment with functions over time. Moreover, each function must be tested and documented. It is best practice to create a separate environment that houses not only the functions but also their associated documentation and testing. This environment is a package (Figure 2.3.1). In addition to the code management benefits, a package can also be shared. In this way, the effort put into the development of a function not only benefits the author but also contributes to a collaborative ecosystem where users can employ the code from different authors, and they even can contribute to improving the code.

In addition to the structure that organizes the functions and their related elements, there is also the need for a structure that orchestrates the different workflows or modules. As functions are modular blocks within code structures or files, these files can also be organized within a project into modular blocks, where each block takes care of one specific task (Figure 2.3.2). In that way, a main script coordinates the overall flow of processes and data. This main script can call helper scripts, where each one is in charge of one sub-process.

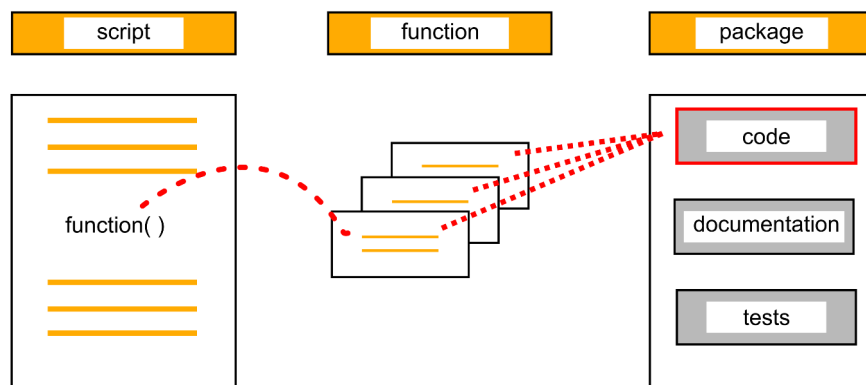


Figure 2.3.1: Functional programming and packages

The code of the functions is stored in a package, together with its documentation and testing. These functions can be called in a script by specifying the name and the arguments.

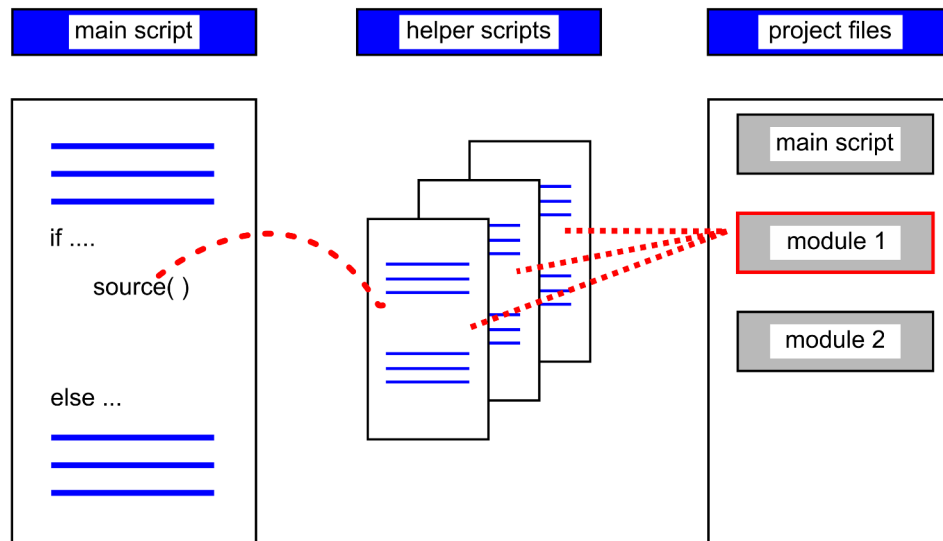


Figure 2.3.2: Modular programming at project level

The project is organized in modules, with each module dedicated to a specific task. These tasks are further divided into sub-sections, including the pre-processing, processing and post-processing of data. Pre-processing prepares the data for processing, processing is responsible for solving the task, and the post-processing stores or visualizes the results.

2.4 The Replicated Algorithm

The aim of the replicated algorithm is to detect canopy layers in a forest point cloud. It is divided into three modules: the stratification module, which stratifies the point cloud into canopy layers. A raster module that stores and retrieves information from the raster surface layers. And the evaluation module, which compares the raster surface layers with the tree metrics from the inventory data.

These three modules are presented in the following sections. Technical details and graphics of the stratification and evaluation module can be found in Appendix A, code examples of functions in the stratification module in Appendix E. Section 2.4.1 starts with an overview of what stratification is, how the stratification module solves its tasks and what the processing results are. Section 2.4.2 is dedicated to the evaluation module, which evaluates the results of the stratification module. The Feature Importance Analysis in Section 2.4.3 provides insights into the patterns that control the accuracy of the results. The tables and descriptions of these patterns can be found in Appendix B.

2.4.1 Stratification

2.4.1.1 - The Stratification Module

The stratification module is the core component of the replicated algorithm and is used to detect canopy layers. Canopies under leaf-on conditions are volumes with high point density. The purpose of the stratification module is to identify these volumes by analyzing the vertical distribution of points in the point cloud, as shown in Figure A.3.

To accomplish this task, the module uses a cylindrical kernel that systematically traverses the specified analysis range according to a grid. See Figure 2.4.1 for the analysis range, and Figure 2.4.2 for the grid. Figure A.2 describes the movement of the kernel. This kernel dynamically adjusts its size to the number of points available for analysis.

In the cylindrical kernel, the first canopy layer is detected considering all points. Hamraz et al. then remove the points from the kernel that belong to this layer, and detect the subsequent canopy layer from the remaining points in the kernel. This process is iterative and stops at a defined height above ground. Two methods of stratification are used in this study: Mode 1 retains all points in a kernel throughout the detection of canopy layers, and Mode 2 follows the approach of Hamraz et al. and removes points in the kernel during the detection of canopy layers.

2.4.1.2 - Pre-Processing of the Input Data

The input data for the stratification module are the point clouds of the four transects within the Bavarian Forest National Park. The spatial extend of the data must be reduced to the processing range of the plot on which the analysis takes place (Figure 2.4.1).

All plots are circular and cover an area of 500 m², which corresponds to a radius of approximately 13 m. Extending the analysis range to a circular area with a radius of 30 m mitigates potential edge artifacts and includes contextual information for the analysis. Since the processing takes place within a kernel of a maximum radius of 6 m, an additional buffer of 6 m is required for the stratification.



Figure 2.4.1: Analysis and processing ranges around a plot

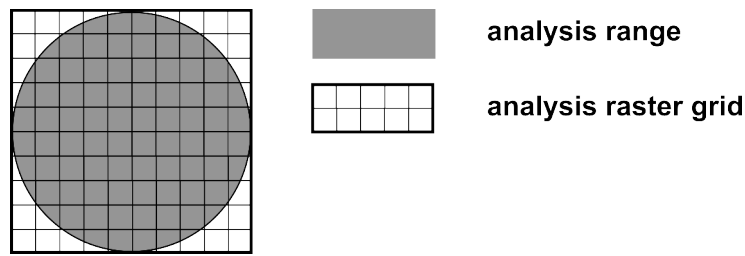


Figure 2.4.2: Analysis raster grid

Once the spatial extent of the point cloud has been reduced, the next step is to normalize the heights of the points. This is done by subtracting the terrain height, which is stored in a Digital Terrain Model (DTM), from the height value. The `normalize_height` function from the `lidR` package [25] generates such a Digital Terrain Model on-the-fly using a knn inverse distance weighting, and normalizes the point cloud. At this stage, the point cloud data is ready for processing.

2.4.1.3 - Output Data Structure

Before processing can begin, the data structure that will contain the processing results must be defined. This is a stack of raster layers, with each layer having a resolution of 1 meter and dimensions of 61 by 61 cells (Figure A.1). Each cell within a raster layer potentially contains one value of a canopy layer boundary. Since each canopy layer has an upper boundary, a lower boundary and a threshold separating one canopy layer from the next, there are three raster layers per canopy layer. The data structure is designed for eight canopy layers, and all these raster layers are combined in one stack. As there are two Modes in the stratification module, there are two stacks of raster layers.

2.4.1.4 - Processing

The processing of the point cloud data follows the grid structure of the output raster layers (Figure 2.4.2), whereby the analysis is carried out in each raster cell that contains points from the point cloud.

To control the scope of the analysis, a cylindrical kernel is used to clip the point cloud to a radius of 6 m around the center of the raster cell. Within this kernel, the total point density is calculated (Figure A.3). Based on this point density, the clipped point cloud is further reduced to the radius defined by the formula of Hamraz et al., which is six times the average footprint. The average footprint is the number of points divided by the horizontal area covered by the point cloud. It is calculated as the reciprocal of the square root of point density.

This adaptive kernel balances the number of points considered for the density analysis: if there are few points, it is enlarged; if there are many points, it is reduced. Since the resolution of the raster grid is one meter, the minimum kernel size in this study is set to a radius of 1.5 m to ensure overlap with neighboring cells. The maximum kernel size is limited to a radius of 6 m, which corresponds to a point density of 1 point per square meter. After this selection, the points within the kernel are filtered to a minimum height of 3 m above ground to exclude undergrowth vegetation, and passed to the density function.

The `density` function from the `stats` package produces the density curve that is analyzed in the next step (Figure A.3). The function uses a Gaussian kernel with a bandwidth of 3 for smoothing. This increases the sensitivity for canopy layers compared to the approach of Hamraz et al. who chose a bandwidth of 5. The minimum height of a canopy layer is set to 3 m.

When analyzing the density curve, the first derivative is used to determine the maxima, and the second derivative is used to determine the start and end points of these maxima. These start and end points are the upper and lower boundaries of a canopy layer, and their values are stored in the respective raster layer. The threshold that separates two canopy layers is the middle between two canopy layers.

2.4.1.5 - Key Terminology

The values of the upper and lower boundaries of a canopy layer, and the thresholds that separate two canopy layers are stored in separate raster layers (Figure A.3). Each raster layer is a surface model. The surface model of a top boundary of a canopy layer is called Canopy Height Model (CHM), its visualization can be found in Appendix D, figures D.1 to D.5. Such a CHM exists for every canopy layer that the stratification module was able to identify. The entire stack of surface models is a canopy model.

2.4.2 Evaluation

The purpose of the evaluation is to quantify the extent to which the canopy model reflects the canopy structure of a forest. In this study, the canopy metrics of individual trees are the reference for this evaluation. These metrics are the coordinates of the canopy base and the tree top from the inventory data.

A tree canopy starts at the canopy base and ends at the tree top or *apex*. The accuracy displayed by the canopy model in these points is an estimate for the model accuracy. It is important to note that in this approach the stratification and the evaluation operate at tree level and therefore a fine resolution is required.

2.4.2.1 - The Evaluation Module

The evaluation module uses the canopy model from the stratification module and the canopy metrics from the inventory data as input. Its first task is to determine the equivalents for the tree top and the canopy base in the canopy model. The tree tops are detected from the surface layers that correspond to the upper boundary of a canopy layer, and the canopy base from the layers that correspond to a lower boundary of a canopy layer.

The second task of the evaluation module is to estimate how well the canopy model fits the canopy metrics from the inventory data by introducing two measures: canopy accuracy and canopy precision. Canopy accuracy refers to the percentage of the canopy height that is detected. Canopy precision is the percentage of the canopy interval that is covered by the canopy model (Figure 2.4.3).

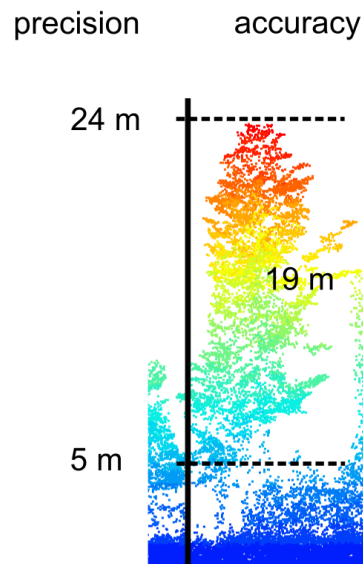


Figure 2.4.3: Canopy accuracy and canopy precision

Definition: Canopy accuracy refers to the percentage of the canopy height, that is detected. Canopy precision is the percentage of the canopy interval that is covered by the canopy model.

Example: In the inventory data, the canopy starts at the canopy base with a height of 5 m above ground and ends at the tree top with a height of 24 m above ground. The canopy height is 19 m. Assuming that the best fitting tree top from the canopy model is at 25 m above ground and the best fitting canopy base at 7 m above ground, the canopy height derived from the canopy model is 18 m.

- canopy accuracy = $18 / 19 = 0.947$ 95 %
- canopy precision = $(24 - 7) / (24 - 5) = 0.895$ 90 %.

2.4.2.2 - Matching Inventory Metrics and Canopy Model

To find the equivalents for the tree top and canopy base in the canopy model, it must be taken into account that the tree top position observed in the point cloud may not match the measured position in the inventory data. Several factors contribute to these location shifts, including a leaning angle of the tree, inaccuracies in the tree position, and shape distortions due to normalization of the point cloud on a slope [30].

Since the canopy model is based on the point cloud, any possible shift of the tree top position in the point cloud leads to a corresponding shift of the tree top position in the canopy model. To account for this possible shift, a kernel is applied to the raster cells of a surface model to detect possible matches with the measured tree top or canopy base (Figure A.4). The size of this kernel is defined by considering the horizontal shift of the tree top from the tree center at a leaning angle of 5 °. The kernel size is further limited in its dimensions, ranging from a minimum size of 3 by 3 raster cells to a maximum size of 9 by 9 raster cells. Within this kernel, the best fit of the surface values to the canopy metrics from the inventory data is defined as the tree top and canopy base of the canopy model.

2.4.3 Feature Importance Analysis

Since the National Park region has characteristic patterns such as altitude gradient, varying successional stages, and diverse forms of land management [21,22], the overall mean of canopy accuracy and canopy precision is not detailed enough to estimate the performance of the model. In order to identify the patterns that most influence the canopy model, the data is grouped according to these distinct patterns and then analyzed with a Feature Importance Analysis.

2.4.3.1 - Pattern Definition and Grouping of the Data

Before starting the analysis, specific indicators must first be defined for each pattern. A comprehensive list and a description of these indicators can be found in Appendix B. The next step is to assign the respective value of these indicators to all plots and trees. The result of this process is a table in which the indicators are listed in columns for both the trees and the plots, and in rows for the individual trees. Based on this table, the data can be grouped according to the different patterns.

2.4.3.2 - Computing the Feature Importance Analysis

Once this table is filled in, the Feature Importance Analysis can be started. The result is a ranking of the patterns based on their importance for the accuracy of the canopy model (Figure 3.1). The `randomForest` function of the `randomForest` package is used to

perform the Feature Importance Analysis. In this analysis, the predictor variable is either the canopy accuracy or the canopy precision. The table of patterns and indicators serves as the response variable. To evaluate the performance and robustness of the random forest model, a 10-fold cross-validation is performed using the `trainControl` function from the `caret` package.

3. Results

Canopy accuracy and canopy precision are the estimators used to evaluate the fit of the canopy model to canopy metrics of individual trees from the inventory data. Of the 2768 trees in the dataset, 1641 were selected for the evaluation. For all selected trees, the mean value in canopy accuracy is 146 %, the median is 110 %. In canopy precision the mean value is 80 %, and the median is 95 % (Table 3.1).

Table 3.1: Canopy model evaluation

	canopy accuracy	canopy precision ⁽¹⁾	canopy precision ⁽²⁾
mean	146	80	87
median	110	95	97
IQR	68	33	13
SD	109	28	24

The values are %, interquartile range (IQR), standard deviation (SD)

(1) - trees from all plots

(2) - trees with canopy accuracy close to 100 % (values between 95 and 105 %)

Analyzing the canopy precision for trees with a canopy accuracy close to 100 %, the mean value is 87 %, and the median is 97 % (Table 3.1, right column). Approximately 20 % of the selected trees are detected with canopy accuracy and canopy precision close to 100 %.

A comparison of the distribution of the data between the three estimators presented in Table 3.1 shows that the canopy precision of the data subset (column on the right) has the smallest difference between the mean and the median. This indicates a relatively symmetrical distribution. In addition, the comparatively small interquartile range indicates a low dispersion of the data within the central 50%, and the standard deviation is the lowest among the three estimators.

However, if all all trees are considered, the data distribution undergoes a notable change and there is a considerable gap between the mean and median values. This indicates the presence of outliers. In addition, the data distribution shows a clear right skew for canopy accuracy and a moderate left skew for canopy precision. For both estimators, the data are widely spread, indicating a high variability.

Looking at the results from the Feature Importance Analysis (Figure 3.1), it is noticeable that:

- The ranking of the patterns is the same, regardless of whether the predictor variable is canopy accuracy or canopy precision.
- The value of the tree top is more important for estimating canopy accuracy and canopy precision than the value of the canopy base.
- The second most important block of patterns are those that were taken into account when selecting the trees to be evaluated.
- At the bottom of the ranking are all the important gradients mentioned in the literature.

When evaluating the performance of the Random Forest model with cross validation, the robustness of the model is weak: about 30 % of the canopy accuracy and the canopy precision value can be explained by the Random Forest model.

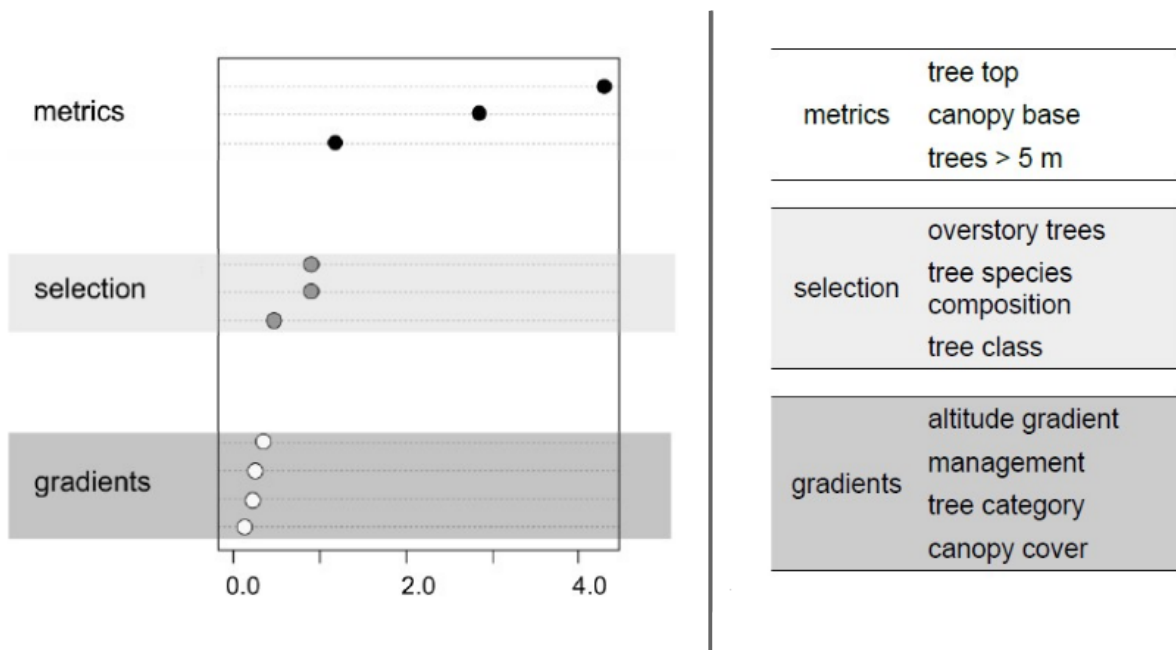


Figure 3.1 Ranking of patterns in the Feature Importance Analysis, grouped into categories.

Table 3.2 Pattern index: a list of the individual patterns. Appendix B provides supplementary information on the patterns and indicators.

4. Discussion

The primary goal of this study was to replicate the stratification algorithm of Hamraz et al. [7]. The secondary goal was to evaluate the achieved stratification using field data. Stratification refers to the detection of canopy layers in forest vegetation. The information of these canopy layers is stored in a canopy model.

The replicated algorithm was applied to a LiDAR dataset from the Bavarian Forest National Park in Germany. The evaluation approach was based on measuring the difference between the tree metrics from the field inventory data and those from the canopy model. Estimators for this difference are canopy accuracy and canopy precision (Figure 2.4.3).

In this discussion, Section 4.1 analyzes the results and provides an insight into the performance of the replicated algorithm. Section 4.2 discusses the influence of the internal structure of the algorithm on these results. Finally, in Section 5, an outlook on the future impact of the replicated algorithm is given.

4.1 The performance of the replicated algorithm

4.1.1 - Canopy Accuracy and Canopy Precision

The estimators canopy accuracy and canopy precision (Figure 2.4.3) are interrelated: an increase in canopy accuracy corresponds to a higher canopy precision. This is due to the fact that the greater the range between the canopy base and the tree top, the greater the probability of accurately covering the height interval. Comparing the values of the two estimators in trees with a canopy accuracy close to 100 % ¹ shows that if the canopy accuracy is correct, the canopy precision also tends to be correct ².

The canopy model fits well to approximately 20 % of the selected trees. This is a low percentage considering that the selection of trees focused on dominant and subdominant trees. The definition of dominant and subdominant trees was determined for each plot based on the existing tree heights, so that these trees or at least their tree tops should be clearly distinguishable. This selection should also exclude overtopped or dominated trees where the tree top is hidden in the canopy layer. Extending the evaluation to such trees could lead to different results with even less alignment between the canopy model and the canopy metrics from the inventory data.

1 values between 95 and 105 %

2 values of 90 % or greater

Two important findings can be derived from the evaluation results: first, the cause of outliers in the data needs to be identified. This analysis will lead to a deeper understanding of where the canopy model is able to detect the forest canopy structure and where it is not. Secondly, the approach of evaluating the canopy model based on canopy metrics of individual trees is questioned. An alternative approach could be an evaluation at plot level, where whole canopy layers are used for the evaluation rather than individual trees.

4.1.2 - Ranking

Analyzing the ranking in Figure 3.1, the most important features for the accuracy of the canopy model are the tree top, followed by the canopy base. This order results from the evaluation approach that is based on these two measures, and from the range of values for both features. While the height of the tree tops ranges from 5 to 48 m above ground, the height of the canopy bases ranges from 0 to 27 m above ground. Therefore, the error that can be committed in falsely detecting a tree top is potentially greater than that of a canopy base.

From third place in ranking, the features are all the patterns described in Appendix B. In third place is the pattern *trees > 5 m*. This could belong to both categories: the *metrics* category or the *selection* category. Putting it into the *metrics* category emphasizes that the number of trees on a plot is of great importance for the accuracy of a canopy model. This result is covered by all the tree detection and benchmark studies mentioned in the introduction. However, the threshold of 5 m was set in this study to select trees that are not classified as shrubs or other vegetation. Therefore, and also due to the similarity of the values it could also belong to the *selection* category.

Nevertheless, the overall message that emerges from the feature ranking is that the evaluation approach and the selection of trees have more influence on the accuracy of the canopy model than the inherent ecological and climatological gradients. This observed bias towards assumptions on data is in contrast to the original algorithm developed by Hamraz et al., which explicitly states that it is free from a priori assumptions on data, and disqualifies the present state of the replicated algorithm.

4.2 The Internal Structure of the Replicated Algorithm

The choice, definition and implementation of algorithm parameters are key elements for the accuracy of a canopy model. The most important parameters are the *tree dominance*, the *resolution*, the *smoothing* and the *evaluation kernel*.

4.2.1 - Tree Dominance

The detection of the tree top and the canopy base in a point cloud is the core issue that determines the accuracy of a canopy model. In ALS point clouds, the tree top can be detected if the tree canopy is in a dominant position, i.e. above or next to other tree canopies. If the tree canopy is overtopped or dominated by other tree canopies, it is difficult to distinguish between tree canopies. These interconnected tree canopies affect the detection accuracy of both the canopy base of the dominant tree and the tree top of the overtopped or dominated tree.

In this evaluation approach, it is essential to consider only trees where a tree top and a canopy base are distinguishable. The tree classes *dominant* and *subdominant* are estimators for these trees. The choice of relative tree height on the plot as an indicator requires a selection threshold that separates the *dominant* and *subdominant* trees from the other tree classes. To obtain a statistically sufficient number of trees per plot, this threshold is subject to an additional condition: at least 20 % of the trees need to be included.

This statistical method is only able to compensate for local differences in stand structure to a limited extent. Therefore, dominated or overtopped trees can also be selected, which in turn reduces the accuracy of the canopy model.

4.2.2 - Resolution

Resolution is a key element in the detection and interpretation of single tree canopies and therefore has a strong impact on the accuracy of the canopy model. In this study, the resolution of the raster grid for the canopy model is 1 m. With this resolution, the canopy surface layers are suitable for the chosen evaluation approach based on individual tree canopies.

However, the high level of detail that can be captured within a tree canopy at this resolution leads to false positives, such as lateral branches being identified as tree tops, and must be reduced by generalization or post-processing. In addition, running the algorithm at a high resolution increases the computational effort of processing and storage, slows down its performance and limits its application on a large scales.

Since high resolution is not required for the detection of coarse structures like canopy layers, the replicated algorithm in its present form should only be used for the detection of canopy layers if other tasks in the workflow such as single tree detection rely on the high level of detail that is generated.

Not only the resolution must be adapted to the task, but also to the size of vegetation and the point cloud density. A low resolution and a low point density smoothens inhomogeneous structures and facilitate the detection of coarse structures such as broad tree canopies. A high resolution and a high point density make structural details visible and facilitate the detection of small or hidden structures such as small tree canopies.

Hamraz estimated in [7,24] the initial point density required for the detection of single trees in each canopy layer. Counting the canopy layers from top to bottom, an initial point density of 30 points per square meter is required for the detection of trees in a second canopy layer and 170 points per square meter for trees in the third layer.

This estimation is based on canopy occlusion which is a function of the vegetation density: the denser the vegetation, the more returns are in the upper canopy layers. As a mean estimate for his research area, Hamraz found out that 86 % of the points are in the first canopy layer, 11 % in the second and 2 % in the third canopy layer [24].

The point cloud density in this study is 35 points per square meter. This proved to be insufficient for the evaluation task: the canopy models of Mode 2³ contains only a few canopy layers, and the number of values in the individual surface layers is not sufficient for a meaningful interpretation or evaluation. Therefore, the evaluation focuses on Mode 1⁴, whereby the interpretability of the canopy layers below the second canopy layer may not be given. A visual comparison of the Canopy Height Models of both Modes can be found in Appendix C.

4.2.3 - Smoothing

The degree of smoothing in the stratification module is a crucial factor in determining the number of canopy layers detected by the algorithm. In this study, a lower degree of smoothing is used compared to the approach of Hamraz et al., which results in a larger number of canopy layers being detected.

This increase in canopy layers means that a tree canopy is divided into different sections. When evaluating the canopy model using existing tree canopy metrics, the overestimation of canopy layers is not an issue. This is because the evaluation module considers only the canopy layers that contain either the tree top or the canopy base from the inventory dataset.

3 Mode 2: the stratification module reduces the point cloud and sets a minimum of 4 points per square meter as an additional condition for defining canopy layers. 4 points per square meter are set as minimum point density for tree segmentation in [24] and as optimum in [22,23].

4 Mode 1: the stratification module does not reduce the point cloud, and does not impose additional conditions for defining canopy layers.

However, when canopy metrics or canopy cover are derived directly from the canopy model, the degree of smoothing must be adjusted to ensure that a canopy layer corresponds to a single tree canopy.

4.2.4 - Evaluation Kernel

The evaluation kernel is designed to account for the possible displacement of the tree top position. Its size is determined by a linear function, that takes into account both the tree height and a fixed leaning angle (Figure A.4). However, it should be noted that this linear function does not consider the variability in the terrain slope or the shape of the tree canopy.

Therefore, the kernel can include values from neighboring trees and select the best fit for the tree top and the canopy base from large tree-height intervals. This falsely increases the accuracy of the canopy model.

5. Outlook

While the main goal of the algorithm is to estimate tree canopies as layers, accurate detection of individual trees is not a primary concern. In this study, the replicated algorithm demonstrates that it is capable of estimating canopy layers and that it has a moderate fit to dominant and subdominant trees. These results were obtained with a point cloud dataset characterized by an insufficient point density for this type of analysis and a forest that is diverse in its structure.

The next step is to make the evaluation module independent of a priori assumptions about trees, apply it to a point cloud with higher point density, and compare in an additional tree detection module whether the replicated algorithm is able to achieve similar results to Hamraz et al. [7] If similar results are obtained using the same tree detection method, this would indicate that the replicated algorithm is in agreement with the approach of Hamraz et al.

While increasing the affinity to the original algorithm is a primary goal in the development of the replicated algorithm, a subsequent benchmarking study could evaluate the overall quality and efficiency in comparison to established and proven algorithms. A very efficient algorithm was developed by Krzystek et al. [9], who used a multi-sensor approach to detect and classify trees in the same region of the Bavarian Forest National Park. Although tree detection is not the primary goal of the replicated algorithm, this would be a valuable next step in evaluating its performance.

Further changes to the algorithm also include the use of processing modules in C++ and code optimizations such as parallel processing and the implementation of multicore architectures. These optimizations are aimed at reducing and distributing the computational effort to make the algorithm scaleable. Once it successfully confirms its functionality and efficiency, it can be prepared for release as a publicly available R-package.

Given the advances in photonic computing and artificial intelligence, these traditional approaches to point cloud segmentation could be replaced by approaches that are directly applicable to raw point clouds. At this stage, an abstraction like the canopy stratification is still necessary for analyzing large point clouds when the data source is LiDAR only.

6. Conclusion

In summary, it can be said that a point cloud density of 35 points per square meter is not sufficient to assess the fit of the canopy model to all trees. The selection of dominant and subdominant trees demonstrates an overall moderate fit to individual tree canopies. In this evaluation, the mean values for canopy accuracy and canopy precision are 146 % and 80 % respectively. Approximately 20 % of the selected trees are detected with a canopy accuracy and a canopy precision close to 100 %. The parameterization of the algorithm and the selection of trees for evaluation had finally a greater influence on the results than the ecological and climatic gradients within the region.

In order to be used for automated detection of forest canopy layers, the replicated algorithm must operate independently of a priori assumptions or specific parameter configurations. Proof of its ability to estimate entire forest canopy layers has not yet been provided. Once these issues are addressed, the suitability of the replicated algorithm for the detection of individual trees can be evaluated by comparing it with the results of Hamraz et al. [7] and by benchmarking it with the results presented by Krzystek et al. [9].

References

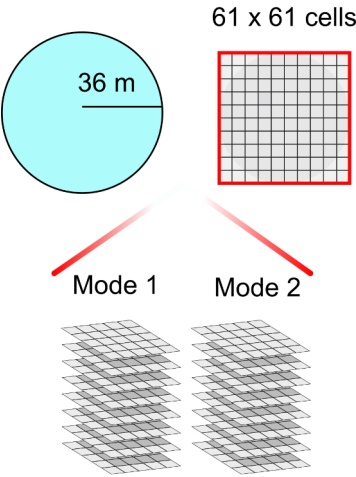
- 1 Latifi, Hooman, Fassnacht, Fabian E., Müller, Jörg, Tharani, Agalya, et al. (2015) 'Forest inventories by LiDAR data: A comparison of single tree segmentation and metric-based methods for inventories of a heterogeneous temperate forest'. *International Journal of Applied Earth Observation and Geoinformation*, 42, pp. 162–174.
- 2 Briechle, S., Krzystek, P. and Vosselman, G. (2021) 'Silvi-Net – A dual-CNN approach for combined classification of tree species and standing dead trees from remote sensing data'. *International Journal of Applied Earth Observation and Geoinformation*, 98(102292).
- 3 Hastings, Jack H., Ollinger, Scott V., Ouimette, Andrew P., Sanders-DeMott, Rebecca, et al. (2020) 'Tree species traits determine the success of LiDAR-based crown mapping in a mixed temperate forest'. *Remote Sensing*, 12(2), p. 309.
- 4 Zhen, Zhen, Quackenbush, Lindi and Zhang, Lianjun (2016) 'Trends in automatic individual tree crown detection and delineation—evolution of LiDAR data'. *Remote Sensing*, 8(4), p. 333.
- 5 Holmgren, Johan and Lindberg, Eva (2019) 'Tree crown segmentation based on a tree crown density model derived from Airborne Laser Scanning'. *Remote Sensing Letters*, 10(12), pp. 1143–1152.
- 6 Dalponte, Michele, Reyes, Francesco, Kandare, Kaja and Gianelle, Damiano (2015) 'Delineation of individual tree crowns from ALS and hyperspectral data: a comparison among four methods'. *European Journal of Remote Sensing*, 48(1), pp. 365–382.
- 7 Hamraz, Hamid, Contreras, Marco A. and Zhang, Jun (2017) 'Vertical stratification of forest canopy for segmentation of understory trees within small-footprint airborne LiDAR point clouds'. *ISPRS Journal of Photogrammetry and Remote Sensing*, 130, pp. 385–392.
- 8 Li, Wenkai, Guo, Qinghua, Jakubowski, Marek and Kelly, Maggi (2012) 'A new method for segmenting individual trees from the LiDAR point cloud'. *Photogrammetric Engineering and Remote Sensing*, 78, pp. 75–84.
- 9 Krzystek, Peter, Serebryanyk, Alla, Schnörr, Claudius, Červenka, Jaroslav and Heurich, Marco (2020) 'Large-scale mapping of tree species and dead trees in Šumava National Park and Bavarian Forest National Park using Lidar and multispectral imagery'. *Remote Sensing*, 12(4), p. 661.
- 10 Vincent, Luc and Soille, Pierre (1991) 'Watersheds in digital spaces: an efficient algorithm based on immersion simulations'. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6), pp. 583–598.

- 11 Dalponte, Michele and Coomes, David A. (2016) 'Tree-centric mapping of forest carbon density from airborne laser scanning and hyperspectral data'. *Methods in Ecology and Evolution*, 7(10), pp. 1236–1245.
- 12 Kaartinen, Harri, Hyyppä, Juha, Yu, Xiaowei, Vastaranta, Mikko, et al. (2012) 'An international comparison of individual tree detection and extraction using airborne laser scanning'. *Remote Sensing*, 4(4), pp. 950–974.
- 13 Vauhkonen, Jari, Ene, Liviu, Gupta, Sandeep, Heinzl, Johannes, et al. (2012) 'Comparative testing of single-tree detection algorithms under different types of forest'. *Forestry*, 85(1), pp. 27–40.
- 14 Eysn, Lothar, Hollaus, Markus, Lindberg, Eva, Berger, Frédéric, et al. (2015) 'A benchmark of LiDAR-based single tree detection methods using heterogeneous forest data from the alpine space'. *Forests*, 6(5), pp. 1721–1747.
- 15 Aubry-Kientz, Méline, Dutrieux, Raphaël, Ferraz, Antonio, Saatchi, Sassan, et al. (2019) 'A comparative assessment of the performance of individual tree crowns delineation algorithms from ALS data in tropical forests'. *Remote Sensing*, 11(9), p. 1086.
- 16 Falkowski, Michael J., Evans, Jeffrey S., Martinuzzi, Sebastian, Gessler, Paul E. and Hudak, Andrew T. (2009) 'Characterizing forest succession with LiDAR data: an evaluation for the Inland Northwest, USA'. *Remote Sensing of Environment*, 113(5), pp. 946–956.
- 17 Li, Qiaosi, Wong, Frankie Kwan Kit and Fung, Tung (2021) 'Mapping multi-layered mangroves from multispectral, hyperspectral, and LiDAR data'. *Remote Sensing of Environment*, 258, p. 112403.
- 18 Silva, Carlos Alberto, Klauberg, Carine, Hudak, Andrew T., Vierling, Lee A., et al. (2016) 'A principal component approach for predicting the stem volume in Eucalyptus plantations in Brazil using airborne LiDAR data'. *Forestry*, 89(4), pp. 422–433.
- 19 Ayrey, Elias, Fraver, Shawn, Kershaw, John A., Kenefic, Laura S., et al. (2017) 'Layer stacking: a novel algorithm for individual forest tree segmentation from LiDAR point clouds'. *Canadian Journal of Remote Sensing*, 43(1), pp. 16–27.
- 20 Grau, Eloi, Durrieu, Sylvie, Fournier, Richard, Gastellu-Etchegorry, Jean-Philippe and Yin, Tiangang (2017) 'Estimation of 3D vegetation density with Terrestrial Laser Scanning data using voxels. A sensitivity analysis of influencing parameters'. *Remote Sensing of Environment*, 191, pp. 373–388.
- 21 Bässler, Claus, Förster, Bernhard, Moning, Christoph and Müller, Jörg (2009) 'The BIOKLIM project: biodiversity research between climate change and wilding in a temperate montane forest – the conceptual framework'. *Waldökologie, Landschaftsforschung und Naturschutz*, 7, pp. 21-34.

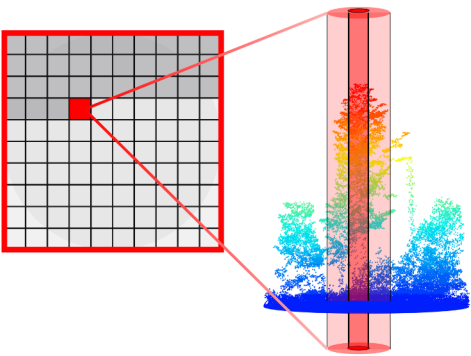
- 22 Hilmers, Torben, Bässler, Claus, Friess, Nicolas, Heurich, Marco, et al. (2018) 'Changes in forest structure in the Bavarian Forest National Park – an evaluation after 10 years of the BIOKLIM-Project'. *Silva Gabreta*, 24, pp. 161–170.
- 23 Kramer, H. and Akça, A. (1995) 'Leitfaden zur Waldmeßlehre'. *Leitfaden zur Waldmeßlehre*, Sauerländer: Frankfurt am Main, Germany, 3rd ed., p. 266.
- 24 Hamraz, Hamid (2018) 'Automated tree-level forest quantification using airborne LiDAR'. *Theses and Dissertations--Computer Science*.69. [online] Available from: https://uknowledge.uky.edu/cs_etds/69 (Accessed 12 Aug 2023).
- 25 Roussel, Jean-Romain, Auty, David, Coops, Nicholas C., Tompalski, Piotr, Goodbody, Tristan R.H., et al. (2020) 'lidR - an R package for analysis of Airborne Laser Scanning (ALS) data'. *Remote Sensing of Environment*, 251, p. 112061.
- 26 R Core Team (n.d.) 'R: a language and environment for statistical computing'. [online] Available from: <https://www.R-project.org/> (Accessed 12 Aug 2023).
- 27 Hijmans, R (2023) 'terra: spatial data analysis. R package version 1.7-41'. [online] Available from: <https://rspatial.org/> (Accessed 12 Aug 2023).
- 28 Murdoch, D. and Adler, D. (2023) 'rgl: 3D visualization using OpenGL'. [online] Available from: <https://github.com/dmurdoch/rgl>, <https://dmurdoch.github.io/rgl/> (Accessed 12 Aug 2023).
- 29 RStudio Team (2020) (n.d.) 'RStudio: integrated development for R.' [online] Available from: <http://www.rstudio.com/> (Accessed 12 Aug 2023).
- 30 Nie, Sheng, Wang, Cheng, Xi, Xiaohuan, Luo, Shezhou, et al. (2019) 'Assessing the impacts of various factors on treetop detection using LiDAR-derived Canopy Height Models'. *IEEE Transactions on Geoscience and Remote Sensing*, 57(12), pp. 10099–10115.
- 31 Jakubowski, Marek K., Guo, Qinghua and Kelly, Maggi (2013) 'Tradeoffs between LiDAR pulse density and forest measurement accuracy'. *Remote Sensing of Environment*, 130, pp. 245–253.
- 32 Wallace, Luke, Lucieer, Arko and Watson, Christopher S. (2014) 'Evaluating tree detection and segmentation routines on very high resolution UAV LiDAR data'. *IEEE Transactions on Geoscience and Remote Sensing*, 52(12), pp. 7619–7628.

Appendix A - Processing Modules

A.1 Data and Data Structures

<p>Figure A.1</p>  <p>61 x 61 cells</p> <p>36 m</p> <p>Mode 1 Mode 2</p>	<ol style="list-style-type: none"> 1 Reduce the point cloud to a circular area with radius 36 and 30 m from the plot center, and normalize the heights (Figure A.1, upper part). The point cloud with a radius of 36 m is the processing point cloud. 2 Generate a raster grid as terra spatial raster object from the point cloud that has a radius of 30 m. The resolution is 1 m, the unit of the cell value is <i>points per square meter</i>. 3 Multiply the raster grid 48 times, name the raster layers, and store them as a nested list object.
<p>Figure description:</p> <p>Upper part, left: the processing point cloud. Upper part, right: the raster grid.</p> <p>Bottom: two stacks with raster layers.</p>	<p><i>Each canopy layer needs 3 raster layers: for the upper boundary, the lower boundary and the layer threshold. The data structure holds 8 canopy layers per stratification Mode. There are 2 Modes. In total there are $3 \times 8 \times 2 = 48$ raster layers.</i></p>

A.2 Iteration in the Stratification Module

<p>Figure A.2</p> 	<ol style="list-style-type: none"> 4 Convert the raster grid to a point geometry, choose the cell centers as reference. <p><i>This eliminates cells with NA values and generates the geolocation for the center of the stratification kernel.</i></p>
<p>Figure description:</p> <p>Left: iteration in the analysis raster layer. Right: The point cloud with the stratification kernel centered at the raster cell center coordinates.</p>	<ol style="list-style-type: none"> 5 At every cell center with a value, reduce the processing point cloud to a circular area with radius of 6 m and calculate the point density. <p><i>The stratification kernel has a maximum radius of 6 m. The <code>density</code> function from the lidR package takes a convex hull to calculate point density. It stays stable across different scales and is more accurate than point densities derived from pixel metrics or a bounding box. Point densities from concave hulls are more accurate than from convex hulls. [26]</i></p>

A.3 Canopy Layer Detection

<p>Figure A.3</p>	<p>6 Calculate the size of the stratification kernel and reduce the point cloud with the radius of 6 m to this size.</p> <p><i>The average footprint is the square root of the point density divided by the point density. Six times the average footprint is the kernel size, which is 3 m minimum and 12 m maximum.</i></p> <p><i>In Mode 1, the size of the stratification kernel is calculated only once, and all canopy layers are detected from this point cloud (Figure A.3, upper part). In Mode 2, the kernel size adapts to the remaining point cloud. This means that the size of the point cloud, which serves as the input for stratification, changes in every canopy layer.</i></p>
	<p>7 Get the XYZ coordinates from the points of this reduced point cloud with the <code>payload</code> function from the <code>lidR</code> package, and keep them as data frame.</p> <p>8 Generate the density curve from the Z coordinate using the <code>density</code> function from the <code>stats</code> package (Figure A.3, upper part, right side). The bins have a width of 0.05 m. The XY values of the function are kept in a data frame.</p>
<p>Figure description:</p> <p>Upper part: on the right the reduced point cloud with stratification kernel, on the left the density curve with the upper boundary value of canopy layer 1,2 and 3.</p> <p>Middle: the red marked raster cell of the current iteration receives the value of the upper boundary of layer 1, 2 and 3.</p> <p>Bottom: storage of all boundary values in the corresponding raster layer and cell.</p>	<p>9 Calculate the first and second derivatives of the density curve and add them to the data frame. Detect where the sign changes in the second derivative with the <code>sign</code> function from the base package. Cluster the sign values, get their start and end position, and select the cluster ranges that correspond to the density maxima. Store them to the corresponding raster layer and cell (Figure A.3, middle and bottom part).</p> <p><i>After completing the iterations over the raster grid, the processing point cloud of one plot is fully analyzed, and the canopy surface layers for that plot are ready for evaluation. In a post-processing step, delete all raster layers that do not contain numerical values.</i></p>

A.4 Evaluation Module - Detection of the Best Fit in the Canopy Model

<p>Figure A.4</p>	<p>10 Iterate through the trees that are selected for evaluation in each plot. Calculate the size of the evaluation kernel from each tree height with the <code>tan</code> function of the base package.</p> <p><i>The leaning angle is 5°. Convert the number of degrees to radians, and use this value in the tan function. The horizontal shift (r) of the tree top is the tree height (z) multiplied by the tangent of 5 degrees.</i></p> <p>11 Select the canopy layers, that correspond to a tree top and a canopy base.</p>
<p>Figure description:</p> <p>Left: the surface layers of canopy layer 1, 2 and 3, with two layers selected (red outline): in this example, the upper boundary surface layer in canopy layer 1 is used to detect the tree top, and the bottom boundary surface layer of canopy layer 2 is used to detect the canopy base.</p> <p>Right: a section from the selected surface layers with the evaluation kernel (light blue cells). The kernel is centered at the XY coordinates of the tree (red color) and has a dimension of 3 by 3 cells.</p> <p>Bottom right: variables for the calculation of the kernel size; tree height (z), radius of the kernel (r), leaning angle (5°)</p>	<p>12 Start with the first value in the kernel, keep it as the best fitting value, compare if the next value fits better, if so, replace it.</p>

Appendix B - Feature Importance

Table B.1 Patterns and Indicators at Plot Level

plot level				
pattern		description	value	categories
altitude gradient	indicator	height of the plot center	< 900	lowland
	unit	m a.s.l.	900 - 1100	montane
	source	Bässler et al. [21]	< 1100	high montane
canopy cover	indicator	area that is covered by CHM layer cells with value	(88, 100]	closed forest
			(57.6, 88]	forest with gaps
	unit	% of the plot area	(32.1, 57.6]	woodland
	source	calculation of Jenks Natural Breaks classes based on all plots	[3, 32.1]	open land
management	indicator	number of plots with the attribute <i>wilding</i>	> 50	wilding
			< 50	management
	unit	% of all plots of a transect		
source	Hilmers et al. [22]			
overstory trees	indicator	tree attribute <i>dominant</i> and <i>subdominant</i>	exact value	no category
	unit	number of trees		
	source	<i>tree class</i> (Table B.2)		
trees > 5 m	indicator	normalized tree height (m)	exact value	no category
	unit	number of trees		
	source	inventory data		
tree species composition	indicator	attribute <i>tree species</i>	(90 - 100]	broadleaf (BL)
			(60 - 90]	BL dominated
	unit	% of broadleaf trees per plot	(40 - 60]	mixed forest
	source	inventory data	(10 - 40]	conifer dominated
			[0 - 10]	conifer

Table B.2 Patterns and Indicators at Tree Level

tree level				
pattern		description	value	categories
tree class	indicator	relative tree height (m) within the plot	n n-1	dominant subdominant
	unit source	class number Scotts rule; calculating optimal number (n) of classes	1 : (n-1)	dominated
tree category	indicator	attribute <i>tree species</i>		broadleaf
	source	inventory data		conifer

Appendix C - Canopy Models from Mode 1 and Mode 2

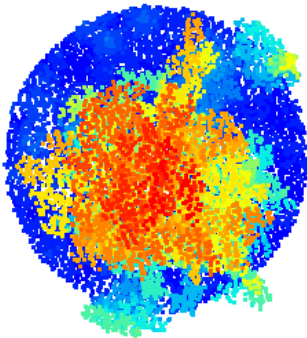
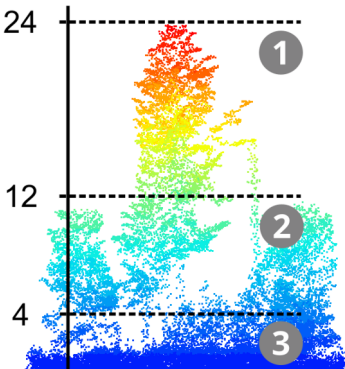
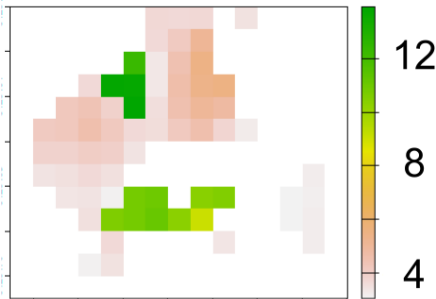
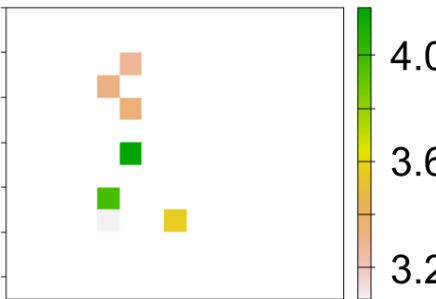
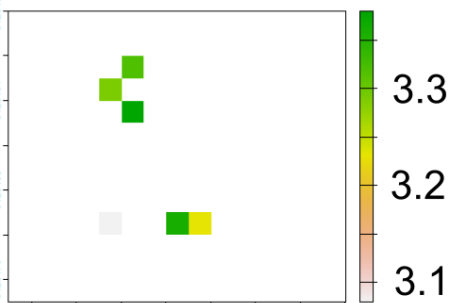
	top view	side view
Transect: 2 Plot: 23 Tree height: 24 m Can. base 5.2 m beech tree living tree	 <p>Figure C.1</p>	 <p>Figure C.2</p>
	Mode 1	Mode 2
canopy layer 2	 <p>Figure C.3</p>	 <p>Figure C.4</p>
canopy layer 3	 <p>Figure C.5</p>	no canopy layers

Figure description:

This figure shows a normalized point cloud (top row) and the two canopy models from Mode 1 and Mode 2 for a tree. The dimensions in the scales of the point cloud and the surface layers are meters above ground, the surface layers are from the upper boundary of a canopy layer. The canopy layers are not post-processed, gaps and outliers remain in the data. The side view of the point cloud (top right) shows the maximum values of the surface layers, the description (top left) shows the information from the inventory data.

Mode 1 from the stratification module detects more canopy layers than Mode 2. In this example, the canopy layer 3 from Mode 1 and the canopy layer 2 from Mode 2 are similar in the number of cells with value, their position and the cell values.

Appendix D - Plot Point Clouds and CHMs

Figures D1 to D4 are presented in the following order:

Figure - A		Figure - B		Figure - C	
subject:	point cloud	subject:	point cloud	subject:	CHM
view:	eagle view	view:	transect W to E	- canopy layer:	1
				- boundary:	upper
				view:	top

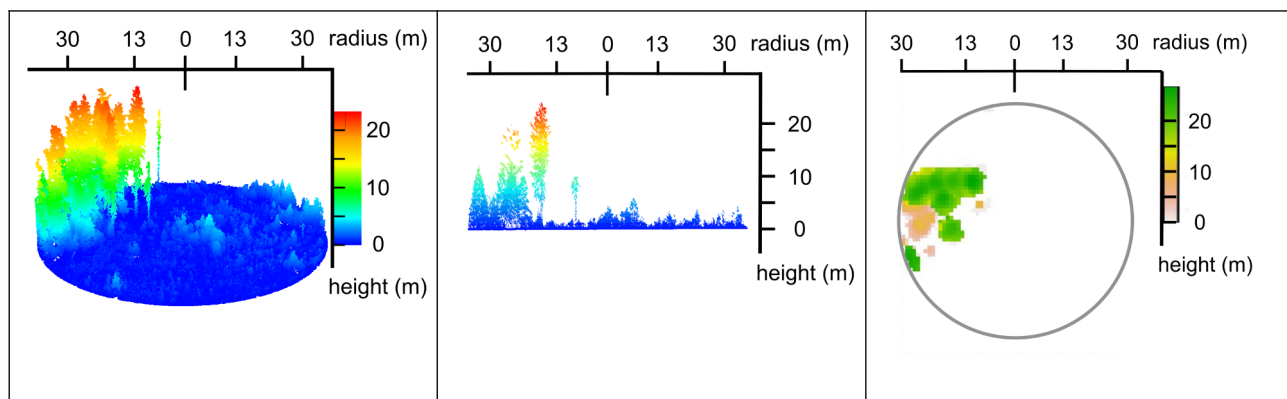


Figure D.1 - open land

- ID: transect 4, plot 70
- altitude gradient: high montane
- species composition: conifer stand

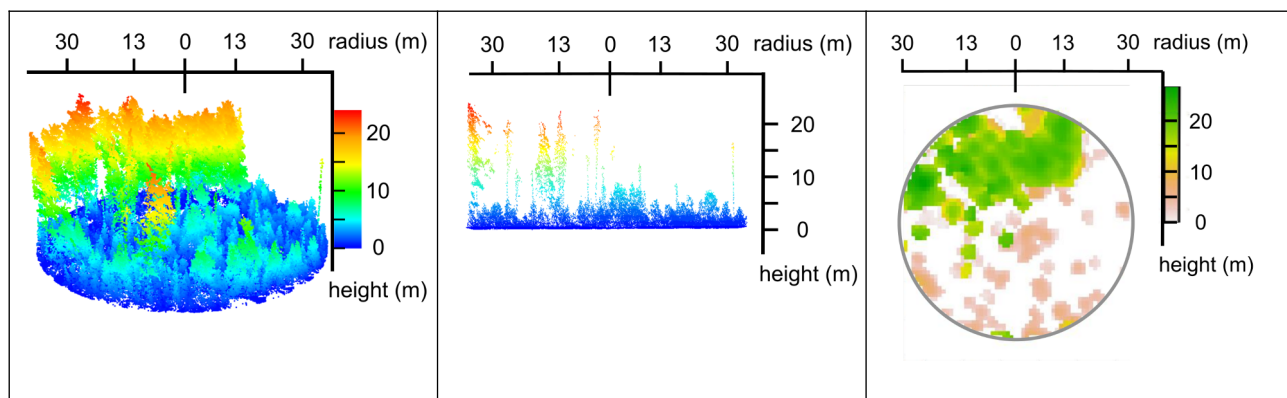
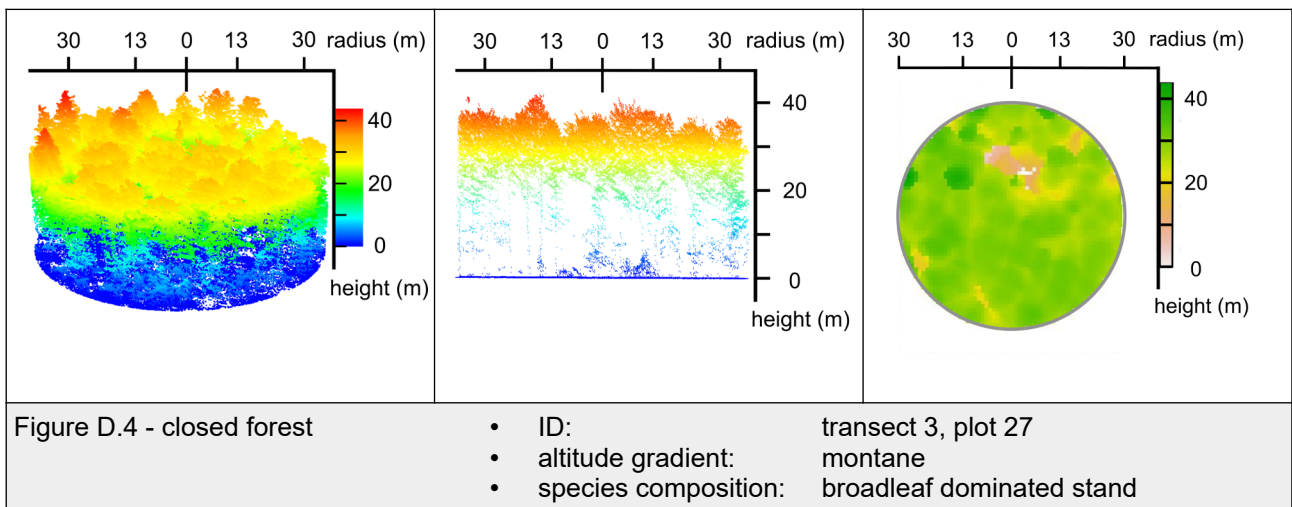
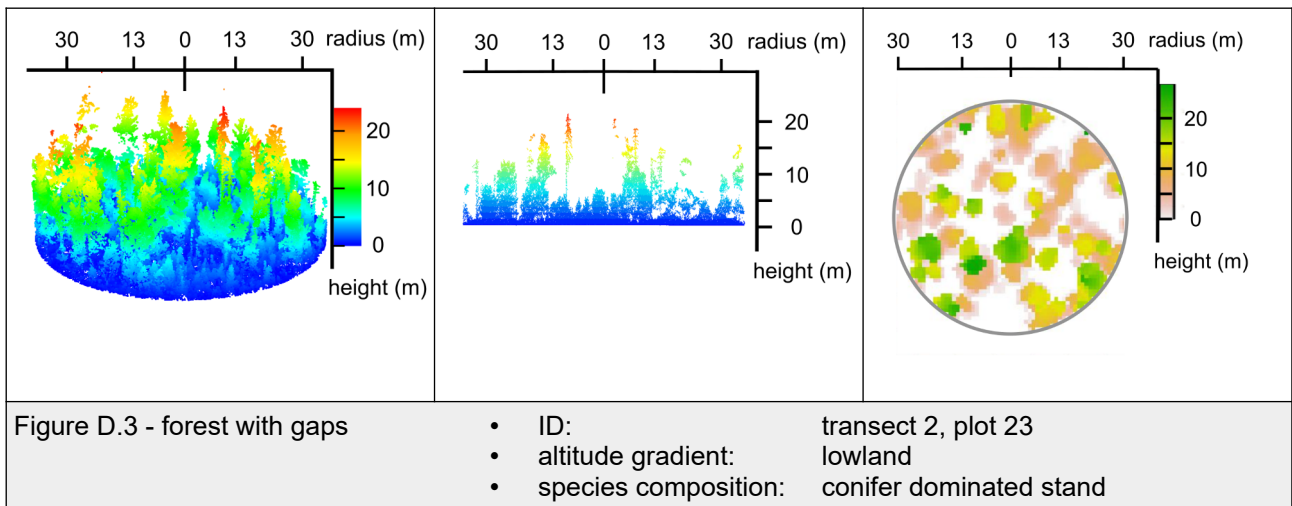
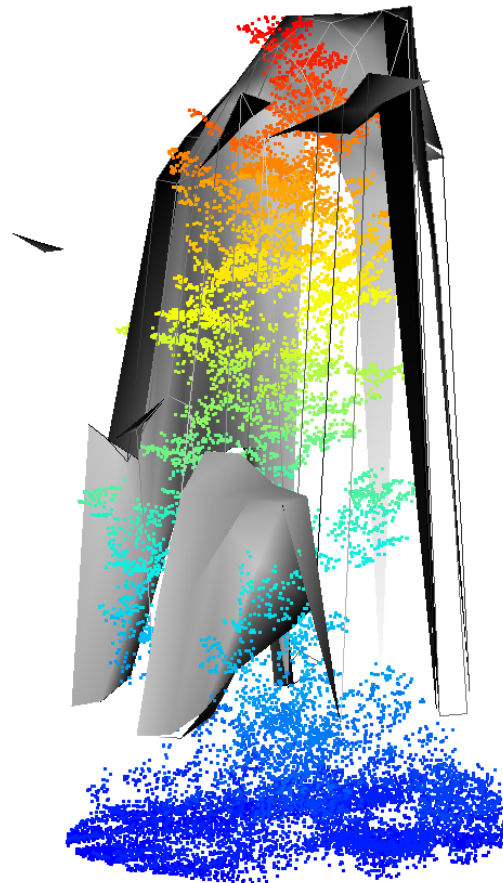


Figure D.2 - woodland

- ID: transect 1, plot 61
- altitude gradient: high montane
- species composition: conifer stand





D.5

3D Canopy Surface Model

Figure D.5 - 3D Canopy Surface Model from the upper boundary of canopy layer 1

Description: The figure shows an overstory tree with two understory trees. The surface layer has the colors gray and dark gray, and the point cloud with the trees is colored according to the height above ground. The deep blue represents points that are close to the ground, the highest points are colored red.

The surface layer reproduces accurately the shape of the trees and even of individual branches. Smoothing creates a small gap between the point cloud and the surface layer.

Appendix E - Code Examples from the Stratification Module

The code presented in this section is not optimized and does not meet the requirements for being officially published within an R-package.

The functions E.1 to E.4 are stored in a locally available package called "forestcloud". Each of these functions contains a basic documentation in the header section and inline comments. In the code, comments start with a hash sign and the text is displayed in green color.

E.1 serves as a wrapper function, i.e. it acts as a structural component that ties together other functions. Specifically, E.1 wraps the functions presented in E.2, E.3 and E.4. The function names from the forestcloud package are visually highlighted in red.

E.1	Canopy Layer Detection
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39	<pre> #' Canopy Layer Information #' #' @description is a wrapper function to gather information about canopy layers #' in a point cloud. Canopy layers are detected by the distribution of point #' density along the z-axis. First it employ the `forestCloud::pcl_roi_flex()` #' function in order to create the density function of the point cloud subset. #' Than it processes the smoothed density function with #' `forestCloud::canopy_layer_heights()` in order to generate the desired #' canopy layer information. #' #' @param las las object, point cloud in the `.las` file format #' @param plot boolean, TRUE prints plots of the density function #' @param x_coord numeric, value of the x coordinate #' @param y_coord numeric, value of the y coordinate #' @param z_thrs numeric, static threshold on the z axis to define the upper #' bond of the point cloud #' #' @return list object, storing variables for further processing #' @export canopy_layers <- function(las, plot, x_coord, y_coord, z_thrs){ if(is.null(las) is.null(z_thrs)) {return()} # store point cloud as data frame input_context_df <- pcl_roi_flex(las, x_coord, y_coord, z_thrs) if(is.null(input_context_df) is.null(z_thrs)) return() # kernel smoothing f <- kernel_smoothing(input_context_df\$Z, plot) # derivatives, canopy layer heights, cat_range cat_range <- canopy_layer_heights(x = f\$x, y = f\$y) cat_count <- as.integer(length(cat_range\$cat)) out_list <- list(cat_count, cat_range, f\$n) return(out_list) } </pre>

E.2 Cylindrical Stratification Kernel

```
01 #' Adjustable Subset of a Point Cloud
02 #'
03 #' @description Defines a circular region of interest around center coordinates,
04 #'   that adapts its radius according to point density measures. The lower the
05 #'   point density, the bigger the radius. The output is generated for point
06 #'   densities >= 4 points per square unit. The measuring units of the
07 #'   `.las`-file and the xyz-axis must be the same.
08 #'
09 #' @param las las object, point cloud in the `.las` file format
10 #' @param x_coord numeric, value of the x coordinate
11 #' @param y_coord numeric, value of the y coordinate
12 #' @param z_thrs numeric, static threshold on the z axis to define the upper
13 #'   bond of the point cloud
14 #'
15 #' @return data frame with XYZ point coordinates
16 #' @export
17
18 pcl_roi_flex <- function(las, x_coord, y_coord, z_thrs) {
19
20   # remove top canopy layer by height threshold
21   las_red <- lidR::filter_poi(las, las$Z <= z_thrs)
22
23   # limit additional canopy information by point density
24   avg_plotPointDensity <- lidR::density(las_red)
25
26   if(avg_plotPointDensity >= 4) {
27
28     # clip the filtered 6m point cloud to kernel size
29     avg_footPrint <- sqrt(avg_plotPointDensity)/ avg_plotPointDensity # Hamraz2017
30     kernel_size <- 6*avg_footPrint
31     if(kernel_size < 1.5) {kernel_size <- 1.5}
32
33     context_las <- lidR::clip_circle(las_red, x_coord, y_coord, kernel_size)
34
35     # store point cloud as data frame
36     input_context_df <- as.data.frame(lidR::payload(context_las))
37     input_context_df <- input_context_df[c("X", "Y", "Z")]
38
39
40   } else {
41     print(paste0("Point density below 4 ",
42               "at center coordinates: ", x_coord, ", ", y_coord))
43     input_context_df <- NULL
44   }
45
46   return(input_context_df)
47 }
48
```

E.3 Smoothing

```
01 #' Kernel Smoothing
02 #'
03 #' @description Wrapper function around `stats::density()` with a custom
04 #'   definition of the function arguments.
05 #'
06 #' @param df_col numeric vector, or column of a data frame object
07 #' @param plot boolean, `TRUE` prints a plot of the density function
08 #' @param kernel character string, indicating the smoothing method. This
09 #'   argument is passed to `stats::density`.
10 #' @param bw character string or integer, the smoothing bandwidth passed to
11 #'   `stats::density`.
12 #'
13 #' @return density function
14 #' @export
15
16 kernel_smoothing <- function(df_col, plot, kernel = "gaussian", bw = 3 ) {
17
18   # discrete values: define interval + steps
19   left <- 0
20   right <- round(max(df_col) + 2, 0)
21   step <- 0.05
22
23   # kernel smoothing
24   f <- stats::density(
25     df_col,
26     kernel = kernel,
27     bw = bw,
28     from = left,
29     to = right,
30     cut = step)
31
32   if(plot == TRUE) plot(f)
33
34   return(f)
35 }
36
```

E.4 Canopy Layer Boundaries

```

01 #' Canopy Layer Heights
02 #'
03 #' @description Returns the layer heights and layer thickness of the canopy
04 #'   layers. The heights are meters above ground, the thickness are meters. A
05 #'   canopy layer is where the point cloud density reaches a maximum value. The
06 #'   first derivative of the density function indicates where a maximum is, the
07 #'   second derivative where this maximum starts and where it ends.
08 #'
09 #' @param x discrete numeric vector, steps of the density function
10 #' @param y numeric vector, values of the density function
11 #' @param pcl_thrs_z positive integer, minimum height (Z) of the point cloud,
12 #'   default are 3 meters above ground
13 #' @param lyr_thrs_m positive integer, minimum height of a canopy layer, default
14 #'   are 3 meters
15 #'
16 #' @return A data frame, where `x.1` is the bottom boundary, `x.2` the upper
17 #' @export
18
19 canopy_layer_heights <- function(x, y, pcl_thrs_z = 3, lyr_thrs_m = 3 ) {
20
21   # create data frame to store the values of f, f1, f2
22   f_df <- data.frame(x, y)
23   f_df[x >= pcl_thrs_z, ]
24
25   # derive function: f1,f2
26   f_df <- deriv_one(f_df, "x", "y")
27   f_df <- deriv_two(f_df, "x", "f1")
28
29   # detect where sign is changing
30   f_df$sign <- c(sign(f_df$f2))
31   f_df$sign_change <- c(0, diff(sign(f_df$f2)))
32
33   sc <- c(1,which(f_df$sign_change != 0))
34
35   # cluster values to canopy layers and add as category to the data frame
36   cat <- seq(letters[1:length(sc)])
37
38   cat_col <- c(diff(sc), length(f_df$f2)-max(sc)+1)
39   cat_col <- rep(cat, times = cat_col)
40
41   f_df$cat <- cat_col
42   f_df$sign_change <- NULL
43
44   # select ranges of interest: negative fluxes
45   f_df_filtered <- dplyr::group_by(f_df, cat)
46   f_df_filtered <- dplyr::filter(f_df_filtered, sign == -1)
47
48   cat_range <- dplyr::group_by(f_df_filtered, cat)
49   cat_range <- stats::aggregate(cat_range, x ~ cat, FUN = range)
50
51   # define minimum layer height: 3 m
52   cat_range <- dplyr::mutate(cat_range, diff = x[,2] - x[,1])
53   cat_range <- dplyr::filter(cat_range, diff >= lyr_thrs_m)
54
55   return(cat_range)
56 }
57

```