MSc thesis in Geomatics for the Built Environment

Accurate, Detailed and Automatic Tree Modelling from Point Clouds

Shenglan Du

2019

ACCURATE, DETAILED AND AUTOMATIC TREE MODELLING FROM POINT CLOUDS

A thesis submitted to the Delft University of Technology in partial fulfillment
of the requirements for the degree of

Master of Science in Geomatics for the Built Environment

by

Shenglan Du

June 2019

ISBN

The work in this thesis was made in the:

3D geoinformation group
Department of Urbanism
Faculty of Architecture & the Built Environment
Delft University of Technology

Supervisors:	Dr. Liangliang Nan
	Dr. Roderik Lindenbergh
Co-reader:	Ir. Agung Indrajit

ABSTRACT

Trees are of great significance throughout the world, both in urban scenes and in natural environments. Models of trees can be widely applied in various fields, for instance, landscape design, geo-simulation, environment modelling, and forestry inventories. Recently, laser scanning technology has been rapidly developed, making it possible to effectively acquire geometric attributes of trees and achieve accurate 3-dimensional tree modelling. Existing studies on tree modelling from laser scanning data are vast. Nevertheless, some works don't ensure sufficient modelling accuracy, while some other works are mainly rule-based and therefore highly depend on user interactions.

In this thesis, we propose a novel method to accurately and automatically reconstruct tree branches from laser scanned points. We first employ the Minimum Spanning Tree (MST) algorithm to extract an initial tree skeleton over the single tree point cloud, then simplify the skeleton through iterative removal of redundant components. A global-optimization approach is performed to fit a sequence of cylinders to approximate the geometry of the tree branches. The results show that our approach is adaptable to various trees with different data qualities. We also demonstrate both the topological fidelity and geometrical accuracy of our approach without significant user interactions. The resulted tree models can be further applied in the precise estimation of tree attributes, urban landscape visualization, etc.

ACKNOWLEDGEMENTS

In this chapter, I would like to give my sincere gratitude to a number of individuals who helped me during my thesis project:

First of all, I would like to thank Liangliang Nan, for offering me the precious opportunity to work in the field of tree modelling. From my side, this topic is very novel, very interesting yet very challenging as well. Liangliang devoted an amount of time discussing with me and guiding me through all stages of this project. Every time I encountered technical or reporting obstacles, he was the one willing to help. Without his patient instructions and warmful support, this thesis work couldn't have achieved to its present form.

I would like to thank Roderik Lindenbergh, Agung Indrajit, Yufu Zang and Kaixuan Zhou for reviewing my work and providing insightful feedback. We had several group meetings together. They took time carefully reading this paper and gave me valuable suggestions within their fields of expertise, which motivated me to dig deeper into this topic.

Special thanks go to Hugo Ledoux, who gave me many practical advises regarding my thesis work. Also thanks Jantien Stoter and the 3D geoinformation group for offering me the opportunity to present my work in the conference FOSS4GNL on 20 June 2019.

A lot of thanks to my parents and my friends, for always being there. Specifically, I would like to thank Hongwei Du, Binian Zhao and Shisi Ruan for constantly supporting me from a remote distance. Thanks to Nikolaos Tzounakos and Vasileios Bouzas for helping me unconditionally with my writing. Also, thanks Meylin Herrera for her warm encouragements.

Finally, thanks to the romantic Chinese kungfu novel, *You Fei*, for accompanying me during this 9-month lonely journey. From the female protagonist Fei Zhou, I learnt how to be a brave and cool girl, ready to face all challenges appearing in life. I'll always remember her saying:

"One day, you will cross the silent river, leaving your quiet hometown and coming to the boundless night sky. You will witness huge mountains to be overturned, and insurmountable seas to dry up. You must remember that your destiny is hanging on the tip of the sword, and you must turn your sword forward. Don't step back, forever."

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACRONYMS

BSG	Branch Strcture Graph	15
CityGML	City Geography Markup Language	8
LiDAR	Light Detection and Ranging	3
LOD	Level of Detail	8
MST	Minimum Spanning Tree	v
PCA	Principal Component Analysis	21
RANSAC	Random Sample Consensus	8
SFM	Structure From Motion	7
VES	Vision Expert System	6

1 INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

Trees are an important component throughout the world. They form and function in natural ecosystems such as forests, and also in human-made environments for instance parks and gardens [Deussen et al., 1998]. Urban scenes without trees or plants are lifeless. Furthermore, satisfying environmental goals always require heavy reliance on vegetation mapping and monitoring [Maltamo et al., 2014]. Models of trees, therefore, have a wide range of applications nowadays.

Applications of tree modelling can be classified into two main categories: First, model artificial trees to be visualized in a digital urban or a natural scene. That is widely applied in urban landscape design, game visualization and movie effects design; Second, reconstruct trees accurately to obtain detailed information of the tree. Important properties of trees, such as the height, the stem thickness, and the crown size, usually play a significant role in environmental-related studies and applications, for example, biomass estimation, forestry inventories and ecological monitoring.

Accurate tree modelling not only enhances the realism within a scene, but also provides promising approaches to scientifically manage vegetations and forests, which will in return contribute a lot to ecosystem protection, resource preservation, preventing degradation, and many other human activities [Ke and Quackenbush, 2011]. Hence, conducting researches in accurate tree modelling is necessary and of great importance to modern society.

The traditional way of measuring trees is to manually conduct fieldwork, which is usually expensive and time-consuming [Hyyppa et al., 2001]. With the development of digital image processing technologies, researchers have tried to conduct tree analysis and modelling from photographs [Reche-Martinez et al., 2004]. A large amount of tree modeling researches based on images have been developed in the past decades [Shlyakhter et al., 2001; Reche-Martinez et al., 2004]. Although these works are capable of producing realistic tree models which can be visualized from multiple views, they don't aim to reconstruct accurate tree branch structure or geometry. Modelling trees from photographs still remains a challenging problem due to the complexity of the modelling process [Guo et al., 2018].

In recent decades, laser scanning technology has been widely used in environmental- and forestry-related studies. As measurements from laser scanning are able to achieve millimetre-level of detail from the objects, it becomes possible to directly capture 3D information and rapidly estimate important attributes of the trees [Liang et al., 2016]. With laser scanning it is easy to obtain accurate point cloud data of trees with high quality and sufficient density, thus making it possible to derive explicit tree models.

To achieve accurate tree modelling from laser scanned points, both the branch geometry reconstruction and the tree skeleton reconstruction are required to be taken account of. Works of [Hackenberg et al., 2014] and [Wang et al., 2016] adopted a cylinder-fitting approach to obtain the geometry of tree branches. To further extract the tree skeletal structure, some works employ a rule-based procedural modelling approach to synthesize branches [Guo et al., 2018; Xu et al., 2007], which will generate the tree skeleton with high quality but requires prior knowledge as well as manual parameters adjustments. Instead, works of [Livny et al., 2010] proposed a pure data-driven method to automatically extract the skeleton without requiring additional user interactions. Nevertheless, the botanical fidelity of the reconstructed tree model is not ensured.

Based on previous researches and studies, we aim to develop a robust approach to automatically reconstruct accurate and detailed 3D tree models from point clouds. Because of the high positional precision of laser scanned data, we use point clouds of real-world trees collected from various scanning devices as our inputs to the proposed approach. The outputs are 3D tree models with detailed branch structure. We want to achieve accurate branch reconstruction of the tree, both geometrically and topologically. Finally, the modelling quality will be evaluated and compared to previous works.

1.2 RESEARCH OBJECTIVE

The goal of this research is to develop a method that can automatically reconstruct tree models from input point clouds. The output tree models should have accurate and detailed branch structure. From the reconstructed 3D models the tree parameters such as the tree height and the tree stem thickness can be measured. To achieve this goal, several relevant sub-questions are listed as follows:

- How to extract the tree branch structure from point clouds? Which data structure is most suitable for storing and representing the tree skeleton structure?

- How can tree tranches be effectively represented? How to reliably reconstruct the tree branch geometry from noisy and incomplete point clouds?

- How to convey the realism of the reconstructed models?

- How to evaluate the reconstructed tree models? How can we improve the modelling quality?

1.3 RESEARCH SCOPE AND CHALLENGES

This thesis project focuses on 3D tree modelling from point clouds, aiming to achieve accurate and detailed branch reconstruction of the trees. The reconstruction of tree leaves is therefore out of the scope of this research. Furthermore, this research focuses on modelling and reconstruction of the individual trees and we assume that each tree is segmented out from the point cloud.

3D tree modelling from point clouds is a challenging topic. Compared to other man-made objects such as buildings, furniture, and mechanical parts, trees typically have more complicated appearances and structures. Due to the variety of tree species, branches as well as leaves, it is difficult to achieve accurate and detailed 3D modelling of trees. On the other hand, data quality is also an important issue that affects the results of modelling. Since occlusion cannot be avoided during the tree scanning process, it is common to obtain tree point clouds with missing data and incomplete branches, thus making the problem more difficult to tackle. To sum up, 3D tree modelling is a valuable research problem. However, multiple challenges need to be addressed to solve this problem.

1.4 THESIS OUTLINE

The thesis paper is organized as follows:

- Chapter 2 reviews the related work, starting from the common data acquisition techniques to various researches and studies for tree analysis and modelling. Both image-based approaches and Light Detection and Ranging (LiDAR) -based approaches are presented. In this chapter, we put our most emphasis on 3D tree modelling approaches from laser scanned data as it's the most relevant with our research work;
- Chapter 3 introduces the whole working pipeline for 3D tree modelling. The skeleton extraction and simplification, as well as the geometry fitting of tree branches, are mainly elaborated with algorithms, diagrams, and formulas;
- Chapter 4 presents the implementation details, the modelling results for various trees from different data sources. Also, relevant discussions are provided in this chapter;
- Chapter 5 gives a brief conclusion together with the future work.

2 RELATED WORK

This chapter aims to provide the relevant knowledge of data acquisition techniques as well as available approaches for tree modelling and analysis. The whole chapter is organized as follows: Section 2.1 gives a short introduction of common data acquisition techniques; Section 2.2 covers an overview of existing approaches to model and analyze trees based on digital images; Finally, researches and studies focused on tree modelling from LiDAR point clouds are addressed in Section 2.3.

2.1 DATA ACQUISITION TECHNIQUES

A conventional way to collect forest and plants inventory data is to conduct periodic field works. Important plant attributes (i.e. height, stem location, volume, etc.) are always assessed by manually measuring sample trees or by using personal experience [Hyyppä et al., 2000]. However, pure ground-based field inventories work on collecting forestry information is often very expensive and tedious [Maltamo et al., 2014].

Since the last several decades, remote-sensing technology has been widely exploited in mapping various information on forests and plants [Kamal et al., 2015]. Both satellite sensors and airborne sensors are capable of effectively acquiring digital images with high spatial resolution, and that provides viable data sources and opportunities for researchers to conduct forestry analysis on the individual tree level [Ke and Quackenbush, 2011]. An early application is the visual interpretation of aerial photography in forest analysis, which serves as an alternative to field measurement [Singh et al., 1986; Ke and Quackenbush, 2011]. However, pure visually-based analysis is usually labour and time-consuming. Lately, various semi- and fully-automated algorithms for detecting and analyzing plants have been rapidly developed, which plays a critical role in modern forestry management because of their high capability in obtaining timely, accurate, and complete plants information [Zhen et al., 2016].

In recent years, the LiDAR technology has been widely used in forestry-related analysis and studies. A LiDAR system is a laser-pulsed system, transmitting laser-pulses and thus determining the distance to the object surface (in forestry study the object can be an individual tree or multiple plants) according to the time of travel [Naesset, 1997]. Typically, several laser pulses can be recorder per m2, and that allows a detailed investigation of forest regions and the creation of 3D tree models [Hyyppa et al., 2001]. There are three typical way of LiDAR scanning: airborne scanning, mobile scanning and static scanning (Figure 2.1).

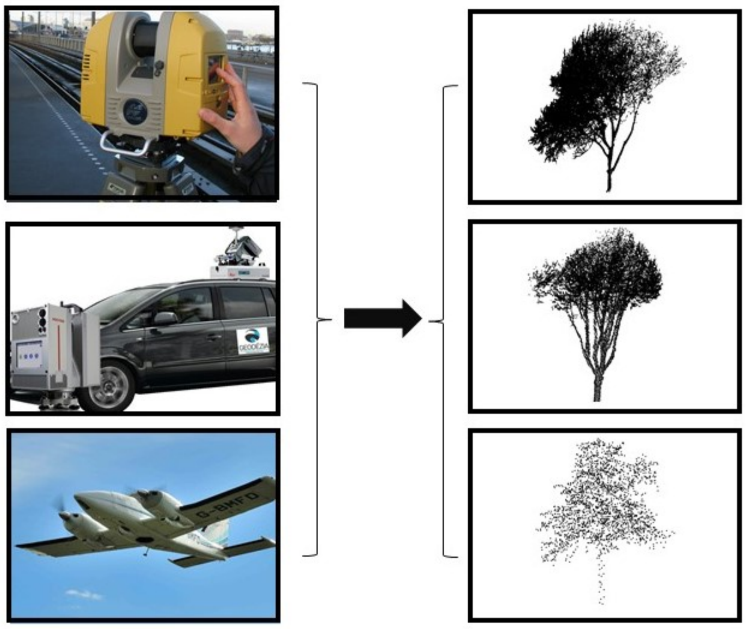


Figure 2.1: Different types of laser scanning. From top to bottom: static scanning, mobile scanning and airborne scanning.

LiDAR measurements can provide points data with high sampling quality, which are widely applied in further researches such as tree diameters and height estimation [Wang et al., 2016], tree canopy analysis [Brandtberg et al., 2003], the identification of tree species [Holmgren and Persson, 2004], etc.

2.2 IMAGERY-BASED TREE ANALYSIS AND MODELLING

Researches and studies in automatic tree analysis and modelling from digital imagery date back to the mid-1980s [Ke and Quackenbush, 2011]. In the early 1990s, [Pinz, 1991] presented the Vision Expert System (VES) to automatically identify tree locations in color photographs. First, the tree crowns are identified by finding circular objects with uniform brightness. To avoid too many texture information which may cause overwhelming local noises within one crown, a low-pass filter is applied before the circle-searching. This work can roughly identify the tree locations and extract the tree crowns' size, nevertheless the detection result is highly influenced by the data noises.

Pollock [1998] developed a model-based method to detect and delineate individual tree crowns. By utilizing high-spatial-resolution optical images as the input data, the work constructs a synthetic image model of the tree crown, which is defined from both geometric and radiometric aspects (i.e. the crown envelope, the sensor irradiance, etc.). Then a matching process is conducted between the images and the synthetic image model instances to recognize the tree crown objects. User-generated data is additionally involved in the data training process.

Besides analyzing the tree stem location and detecting the tree crown, digital images can also be used to reconstruct tree models. In 2001 Shlyakhter et al. utilized images to guide its procedural tree modelling process. The input images are manually segmented into the tree and the background, and then a visual hull is built to approximate the shape of the tree. By finding the medial axis [Ogniewicz and Kübler, 1995] of the tree's visual hull, the plausible tree skeleton is constructed. Then they apply an open L-system [Honda, 1971] to grow the tree and synthesize small branches on top of the main skeleton. Figure 2.2 shows the pipeline of this work.

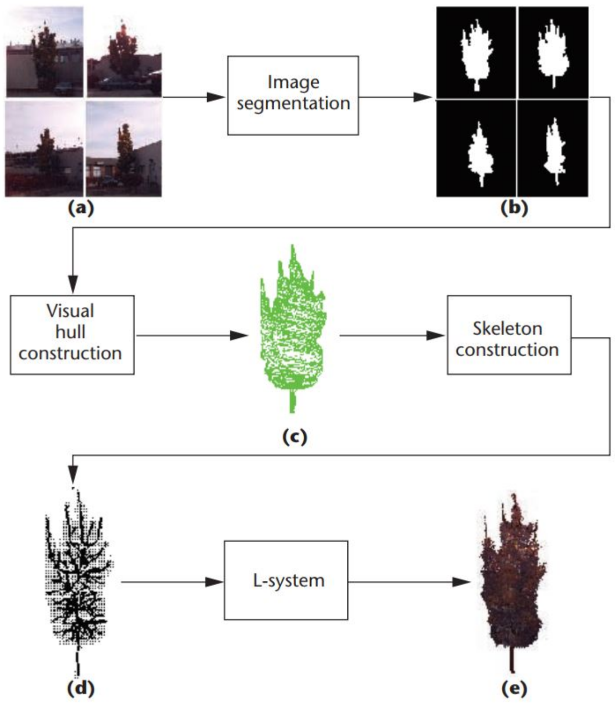


Figure 2.2: Procedurally reconstructing tree models from images [Shlyakhter et al., 2001].

Reche-Martinez et al. [2004] described a volumetric approach to visually reconstruct trees from multiple views. First, a tree is considered as a volume with opacity and colour values. With the opacities estimated through an optimization method, they assign corresponding textures to each pixel in the multiple viewing billboards. This method can realistically model the tree appearance from various viewpoints. However, it doesn't provide accurate information on the tree diameter, the tree volume, etc.

While the works mentioned above can produce impressive modelling results, they don't aim to reconstruct explicit branch or leaf geometry. To solve this limitation, Quan et al. [2006] proposed a points-aided algorithm for modeling trees directly from images. The tree is captured from different views and the corresponding 3D point cloud is generated by applying the Structure From Motion (SFM) technique. Then the user needs to segment the 3D points into individual leaves, which will be used to form the generic leaf model and further model all other visible leaves. The whole methodology is delineated in Figure 2.3. Compared with the previous works, this approach can produce editable tree model which closely resembles the real tree. Nevertheless, it involves tedious user interactions.

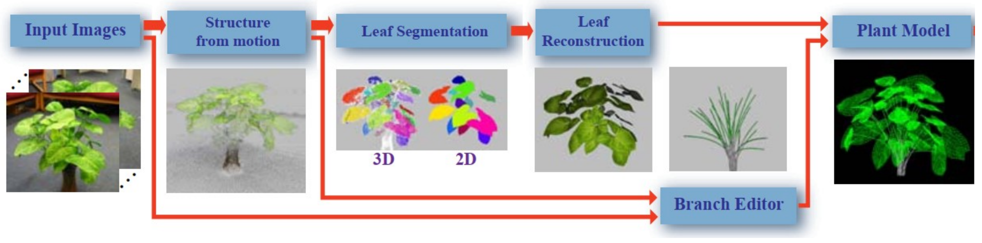


Figure 2.3: Plant modelling from multi-view images [Quan et al., 2006].

2.3 LIDAR-BASED TREE ANALYSIS AND MODELLING

LiDAR is a promising data source for forestry and vegetation studies. Traditionally, it is difficult to extract accurate and detailed information of the trees and vegetation without moving to stereo acquisition. Compared with that, LiDAR actively emits energy and then receives the signals reflected from a target, and thus can directly capture the 3D information of objects [Zhen et al., 2016]. LiDAR technology makes it possible to conduct automatic forestry inventory and tree analysis. Tree attributes (i.e. tree height, stem thickness, volume, etc.) can be easily measured from the laser scanned data [Olofsson et al., 2014]. Moreover, by applying LiDAR technology we are able to acquire highly dense point clouds, which lays the foundation for accurate tree reconstruction and modelling. This section provides vast researches and studies in LiDAR-based tree analysis and modelling, which are closely related to our work and thus we put most of our emphasis on this topic.

2.3.1 Tree analysis from point clouds

Holmgren and Persson [2004] utilized high-density airborne laser scanner data to identify species of individual trees. This work uses field measurements for training and further validation of the classification. Tree point clouds obtained from laser scanning are pre-segmented. Then they are classified according to several characteristics which include the tree height, the crown area, and base height of the crown. The classification results demonstrate high accuracy. Nevertheless, sufficient prior knowledge is required to support tree species detection.

Hyyppa et al. [2001] proposed a segmentation-based method to estimate the standard attributes for individual trees within a region. Raw point clouds obtained from airborne laser scanning are pre-classified into terrain points and vegetation points. Then tree points are segmented using region growing algorithm. Standard tree attributes such as the tree height, the tree crown size and tree volume are thus computed using a series of botanical rules. This work can roughly estimate the individual tree attributes in a large region. However, the estimation results are not very accurate.

Olofsson et al. [2014] proposed an approach to accurately measure the tree stem and the tree height from terrestrial laser scanned data. The tree points are classified into tree stem and tree crown by applying cross-section circles searching. With the pre-classified points, the tree height is estimated from the ground to the crown; stem diameter of the tree is estimated by applying the Random Sample Consensus (RANSAC) algorithm.

2.3.2 "Icon" trees modelling

Currently, the most common approach is to model the trees as icons, which is easy to generate and maintain. Gobeawan et al. [2018] constructed 3D tree models in the format of City Geography Markup Language (CityGML) for the virtual Singapore city (Figure 2.4). In this work, they use the official tree database to determine the locations of trees within Singapore. The laser scanned data are processed to measure the semantic information of individual tree properties (i.e. growth space, branch sizes, crown sizes and shapes, etc.). The trees are modelled in various Level of Detail (LOD). LOD1 is a simple 3D height representation, and LOD2 is a sphpere-crown and trunk representation. This work enables modelling 3D trees dynamically on a large region. However, the reconstructed tree models are not accurate or detailed enough.

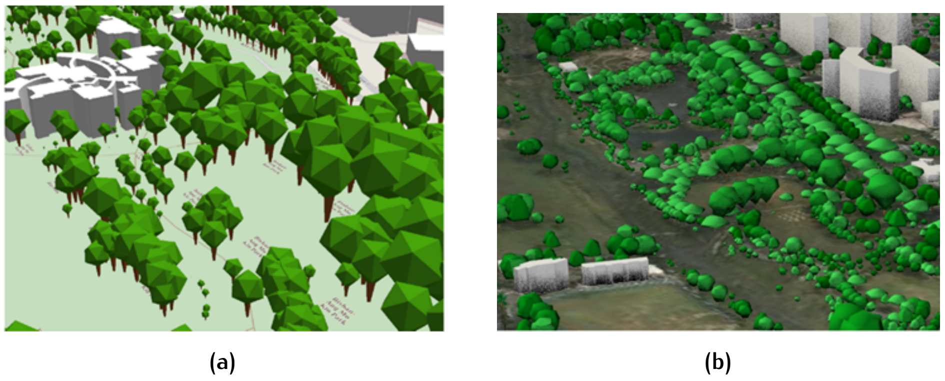


Figure 2.4: Tree models for the Virtual Singapore [Gobeawan et al., 2018]. (a) LOD1 model. (b) LOD2 model.

2.3.3 Tree skeleton extraction

As the "icon" tree models are constructed mainly for visualization, it's difficult to extract useful or measurable information such as the branch structure and the explicit tree skeleton from those models. Compared with methods that directly model trees as icons, a more accurate approach is to reconstruct the tree skeleton from laser scanned point clouds. According to [Cornea et al., 2007], curve-skeletons are 1D structural representations of 3D objects, which is useful for many analysis and visualization tasks such as navigation, animation, model reduction and simplification.

Existing literature on tree skeleton extraction is vast. In 1999, Verroust and Lazarus proposed an algorithm to extract consecutive skeletal curves from scattered points collection. First, they build a neighbourhood graph for the points, where a shortest-path distance map is then computed from a given source vertex. Vertices within the graph are assigned to different levels according to their proximity to the source vertex; And based on the levels the skeletal curves are extracted. The whole extraction process is delineated in Figure 2.5.

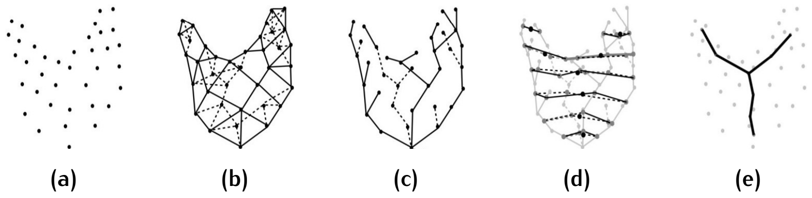


Figure 2.5: Extracting skeletal curves from 3D points [Verroust and Lazarus, 1999]. (a) Scattered points. (b) Neighborhood graph. (c) Shortest path map. (d) Levels. (e) Skeleton.

An alternative for the shortest-path method is the Medial Axis representation for the 3D object skeletal structure, proposed by Dey and Sun [2006]. In this work, the skeleton is defined by the medial axis, which lies within the middle of the object. As illustrated in Figure 2.6, the equidistant 2D surface is first computed and later thinned to a 1D skeleton. Compared with the graph representation proposed by Verroust and Lazarus, this work highly relies on the quality of the input data and therefore is sensitive to noises or irregularities of the input point clouds.

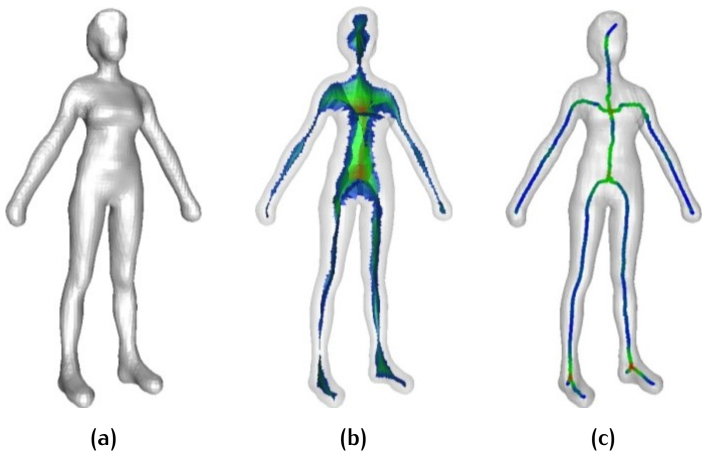


Figure 2.6: Extracting skeleton using Medial Axis function [Dey and Sun, 2006]. (a) Original model. (b) Medial axis surfaces. (c) Thinned skeleton curves.

Following the work by Verroust and Lazarus, Xu et al. [2007] described a graph-reduction method to model the main tree branches from laser scanned data. This work derives the main skeleton by triangulating the input points, finding the shortest path from the root vertex to the rest vertices, quantizing the points into bins based on the length of their paths, and extracting the main skeleton accordingly (Figure 2.7). Additionally, small branches and leaves are synthesized to form the crown geometry.

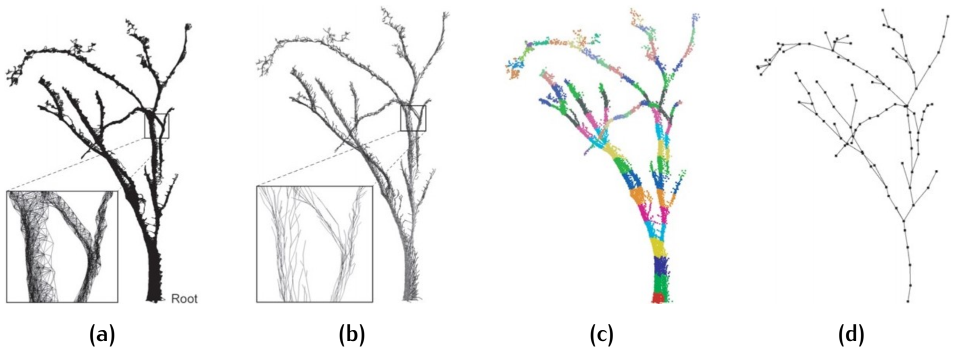


Figure 2.7: Extracting main branch skeleton [Xu et al., 2007]. (a) Triangulation. (b) Shortest-path graph. (c) Quantized bins. (d) Main skeleton.

Instead of extracting skeleton curves directly from point clouds, Bucksch et al. [2009] applied another method, organizing points into the octree structure and generating the skeletal curves from the octree cells. This approach has good performance in terms of computation efficiency. Nevertheless, the results may turn out to be unreliable when the branches don't have uniform spatial distributions.

Similarly, Yan et al. [2009] also applied a cluster-based approach to generate the tree skeleton. The K-mean clustering algorithm is utilized to segment the input point cloud into different branch clusters. Then the branches are identified through the cylindrical components searching, where a neighbourhood graph is built and the skeleton structure is extracted accordingly. This method is effective and fully automatic but is not robust enough to issues such as holes or missing data due to occlusions. The reconstruction method is revealed in Figure 2.8.

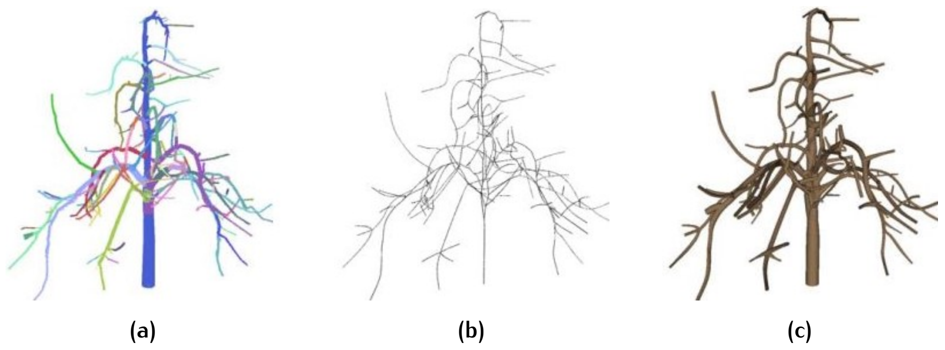


Figure 2.8: Extracting the skeleton from K-means clustering [Yan et al., 2009]. (a) Detected branches. (b) Skeleton. (c) Lofting results.

Livny et al. [2010] introduced a novel approach which computes the minimum spanning graph from the point clouds to obtain an initial skeletal structure of the tree (Figure 2.9). Furthermore, several global optimizations are applied to remove noises and better model the tree branch structure. This method gave us a lot of inspiration, from which our research work will be mainly based on. One significant strength of this method is that it's capable of reconstructing the topological structure of the tree branches from pure point clouds, without requiring additional user interactions or imageries. Nevertheless, this approach still has multiple drawbacks. One limitation is that it cannot provide reliable results when it comes to missing data issues. Moreover, it doesn't guarantee a geometrically-correct tree model, i.e., the output model doesn't fit well to the input point cloud.

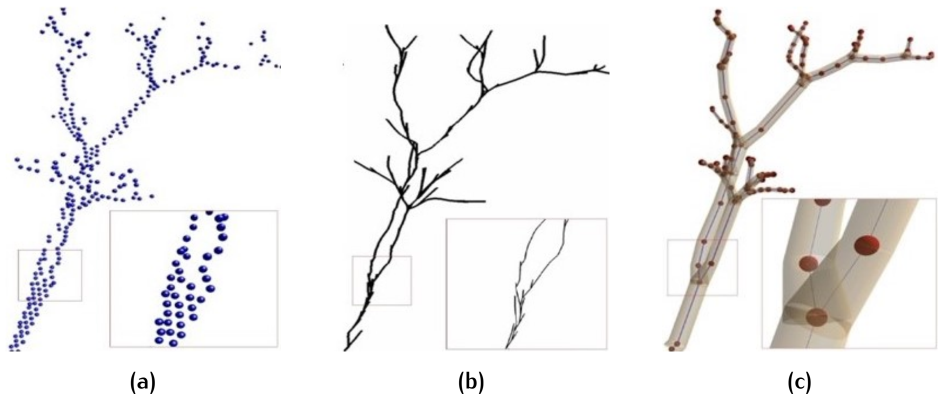


Figure 2.9: Reconstruction of tree branches [Livny et al., 2010]. (a) Input points. (b) Skeleton obtained from minimum spanning tree. (c) Inflated branches.

Most of the approaches introduced are primarily driven by data. Besides that, Guo et al. [2018] applied a rule-based approach to model the tree branch structure. The input data are tree point clouds obtained from stereo tree images. A parametric plant representation is proposed to simulate the tree growth progress and synthesize the 3D skeleton from the point cloud. This method achieves high modelling quality by combining the data-driven reconstruction with tree growth modelling. However, prior knowledge on the tree growth parameters, for instance, the branch rolling angle and the growing unit, requires to be specified by users. Figure 2.10 indicates the important parameters involved in this work.

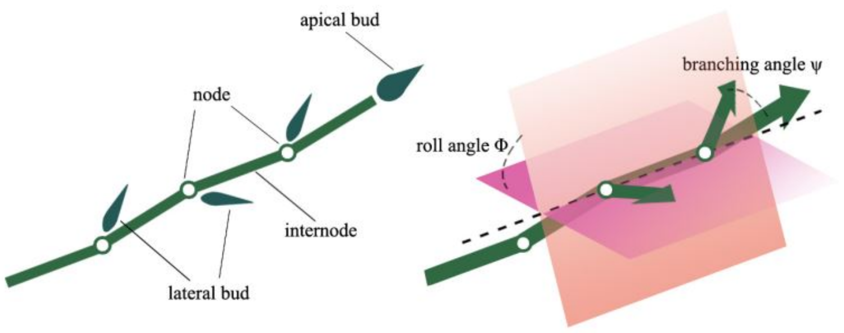


Figure 2.10: Parameters involved when synthesizing the tree branch structure [Guo et al., 2018].

Reconstruction of tree skeletons has many different applications. In 2018, Calders et al. applied a skeleton-based approach to reconstruct forest stands from point clouds, which is later exploited in radiative transfer modelling and simulation. They developed a semi-automatic workflow which enables the segmentation of tree point clouds and the cylinders fitting along with tree skeletons. Furthermore, the laser scanned data is matched with traditional census data to determine the individual tree species and allocate corresponding radiometric properties.

Another application is to investigate the branch architecture of the trees. Lau et al. [2019] collected laser scanned data from several tropical trees. By reconstructing the tree skeleton and branch geometry, the metabolic scaling factors such as the branch radius scaling ratio and length scaling ratio are estimated. Those estimations are later compared with the real values manually measured from the same trees. Results show that estimated metabolic factors of the trees are similar to real values, indicating that the skeleton-based method can be applied in statistical studies of trees and other vegetations.

2.3.4 Branch geometry reconstruction

Having obtained the explicit tree branch structure, researchers are considering to accurately model the tree branch geometry from laser scanned data. When it comes to this topic, the cylinder-fitting approach becomes a mainstream strategy in works of literature [Wang et al., 2016].

In 2014, Hackenberg et al. described a method for fitting cylinders into a tree point cloud. In this work, cylinders are stored as a hierarchical data structure, which enables parent-child neighbour relations. The propagation direction of the tree is also taken into consideration. By applying the hierarchical-cylinder structure, this work can efficiently extract different tree components (for example the stem or a single branch). Nevertheless, this approach is not fully automated, as tree extraction and pre-processing are performed manually.

Raumonen et al. [2013] proposed another method for constructing precise tree models efficiently from point clouds. This approach is based on a step-by-step collection of small connected surface patches, which are extracted from local subsets of the input point cloud. The neighbour-relations and geometrical properties of these patches are extracted and used to segment the point cloud into branches, after which a set of cylinders are used to model the branches. One drawback of this method is that it requires a high quality of the input point clouds. An individual tree often needs to be scanned multi-times from various viewpoints to ensure a good output model.

As most of the methods focus on the tree modelling in a flat area, the work conducted by Wang et al. [2016] paid attention to the tree stem reconstruction in landslide-affected areas, where the terrain is usually steep and the tree stems often grow in an irregular direction. In their work, the tree stem is modelled by fitting a series of cylinders using a RANSAC-based approach. The estimated stem parameters from the reconstructed trunk have various applications (i.e. biomass computation, tree growth estimation, etc.). Figure 2.11 shows that the trunk cylinder is fitted from input scanned points.

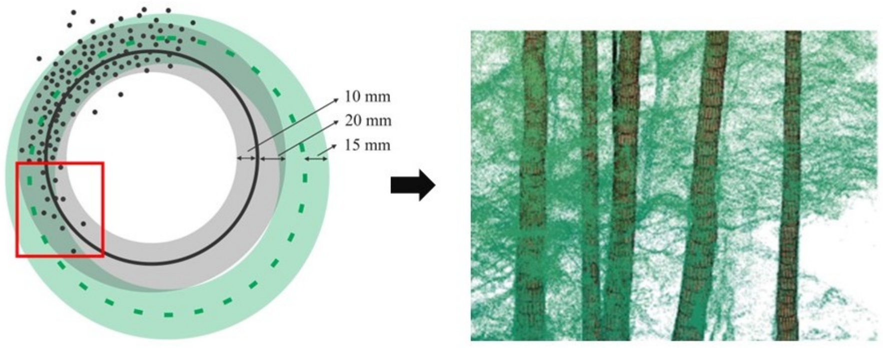


Figure 2.11: Cylinder fitting process for the stems [Wang et al., 2016].

2.3.5 Crown modelling

Having attempted to reconstruct the skeleton structure of tree branches, some follow up works focus on the modelling of tree crowns and tree leaves. Livny et al. [2011] encoded lobe-based tree representations from sample trees and later synthesized them into detailed tree models, resembling the sample tree data (Figure 2.12). This approach requires the prior knowledge of tree species information.

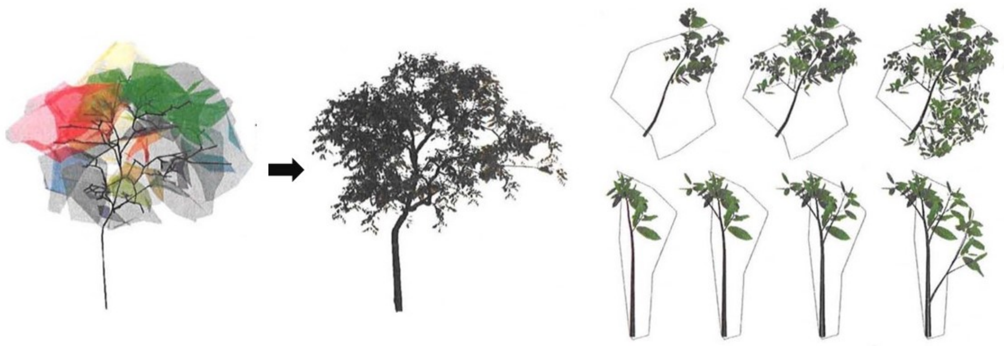


Figure 2.12: Lobe representation of the tree crown [Livny et al., 2011]. Left: dividing the tree crown into and assigning the textured lobes; Right: textured lobes.

In order to enhance modelling realism, Xie et al. [2016] utilized examples of real trees to reconstruct tree models with fine details. 3D real-world trees are captured, sliced, and stored as exemplar tree-cuts (Figure 2.13). Those fine-level geometric details later will be transferred to the reconstructed tree models, enabling that the generated tree models inherit fine realism from the example models.

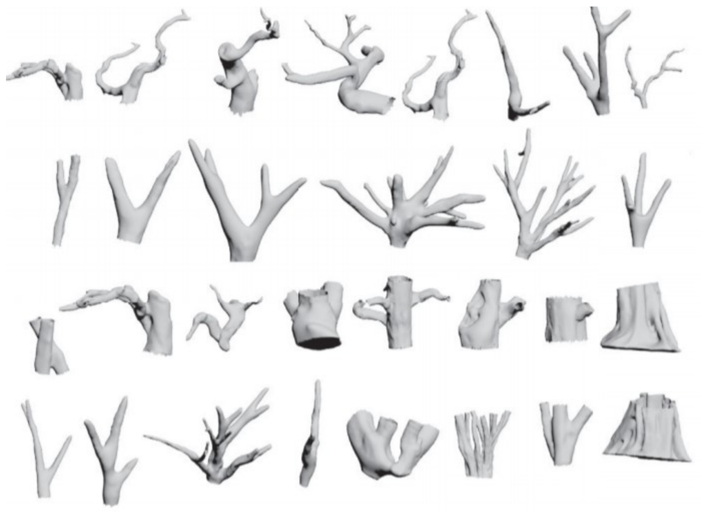


Figure 2.13: Tree-cuts examples with fine details [Xie et al., 2016].

3 METHODOLOGY

Based on the research objective defined in Chapter 1, and the previous researches presented in Chapter 2, we propose our methodology in this chapter. Our input data is the point cloud of a single tree, which typically contains noises and outliers, but is expected to convey the major branch structure of a tree. To reconstruct the tree branch structure in details, we utilize a skeleton-based approach which is inspired by Livny et al.. To improve the topological correctness and the geometrical correctness of the reconstructed 3D model, we further adjust the tree skeleton according to the desired characteristics of the input point cloud. Furthermore, an optimization method is applied to obtain accurate geometry of the tree branches. We aim to produce a fully automatic modelling algorithm, which allows faithful reconstruction for various types of trees while alleviating the users from tedious interactions.

The chapter is organized as follows: Section 3.1 provides an overview of the reconstructed approach; Section 3.2 describes the skeleton-based approach applied to extract the initial tree branch structure; Section 3.3 presents the algorithm we use for simplifying the tree skeleton; Inflation of branch geometry is further introduced in Section 3.4; And Section 3.5 shortly describes the realism enhancing process.

3.1 OVERVIEW

To appropriately represent the tree skeleton, we introduce the concept Branch Strcture Graph (BSG), defined as a spatially connected and directed graph which is acylic [Livny et al., 2010]. The tree BSG has a set of vertices, representing either the tree root, the important turning points, the bifurcations or the leave nodes. The straight edges connecting the BSG vertices represent the tree branches. (Figure 3.3a) gives a complete illustration on the tree BSG.

A good BSG representation of the tree skeleton usually bears several desired characteristics:

- The BSG should fit enough to the input point cloud;
- The BSG should have relatively long branches near the tree root and relatively short branches close to the tree crown;
- The thickness of the branches is highly relevant with their length of subtree.

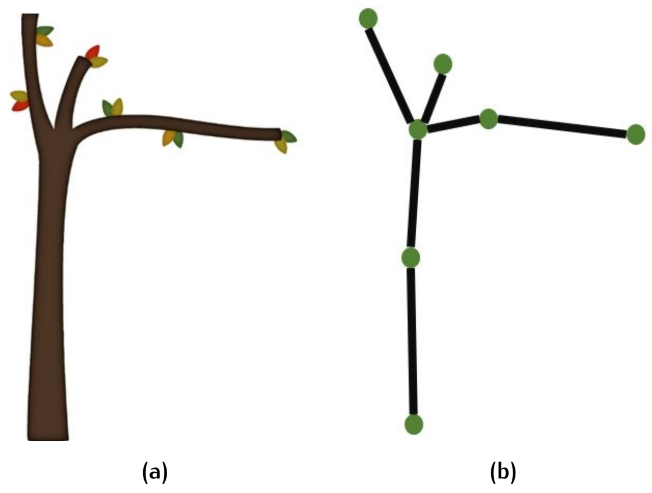


Figure 3.1: Graph representation of the tree branch structure. (a) Real tree. (b) Tree BSG graph.

Our proposed tree reconstruction pipeline is consisted of four major steps, as depicted in Figure 3.2.

- **Initial skeleton extraction**: We triangulate the tree points and apply the shortest-path algorithm to extract the initial skeleton graph. Note that the main-branch points are identified and centralized beforehand to improve the quality of the extracted initial skeleton;
- **Tree skeleton simplification**: The initial skeleton graph is iteratively simplified, resulting in a fine tree skeleton. We simplify the graph by retrieving and merging adjacent vertices if their distance is sufficiently small;
- **Branch fitting**: Based on the reconstructed tree skeleton, we we fit a sequence of cylinders over the input points to approximate the geometry of the branches. We first apply non-linear least squares adjustment to obtain accurate radius of the tree trunk. Then, we derive the radius of the subsequent branches from the main trunk;
- **Adding realism**: Finally, we synthesize random leaves at the end of tree branches and add textures to enhance realism.

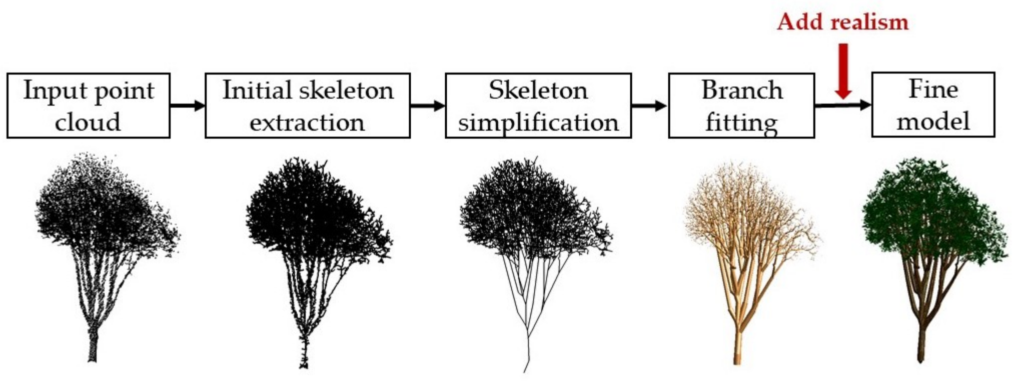


Figure 3.2: An overview of the proposed methodology.

3.2 INITIAL SKELETON EXTRACTION

The very first step of tree modelling is to read input point cloud and extract the initial tree skeleton from points. Points proximal to each other are more likely to belong to the same branch. Based on such characteristic, we construct a MST graph over the input point cloud to represent the tree skeleton. The MST is a weighted graph where the sum of the weights of the edges is minimized. It provides a rough estimation of the intrinsic structure from a given dataset [Zhong et al., 2015]. It's very suitable for representing the tree branch structure, as it constructs the least expensive spanning path for the tree-points.

The initial tree skeleton can be extracted by computing a MST graph from point cloud, which mainly involves two steps:

Firstly, we apply Delaunay triangulation to construct an initial graph from the input points. Delaunay triangulation defines a piecewise linear interpolation function and generates a well-shaped spatial tessellation [Gudmundsson et al., 2002]. It lays the foundation for MST computation as many efficient approaches extract a MST among edges within the Delaunay triangulation of the points [Zhou et al., 2001]. Additionally, Delaunay triangulation helps to complete the missing region or incomplete branches, which will make the skeleton extraction more robust to the input point clouds with poor data quality;

After the initial triangulation graph is constructed, we weight all the edges using their lengths defined in the Euclidean space. Then, we apply the Dijkstra shortest path algorithm to compute the minimum-weighted spanning tree from the Delaunay graph, which will create an initial tree skeleton. The skeleton extraction process is summarised in Figure 3.3.

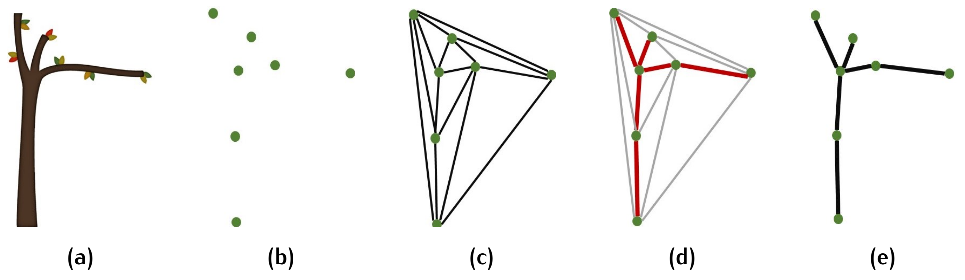


Figure 3.3: Initial skeleton extraction process. (a) Real tree part. (b) Scanned points. (c) Delaunay triangulation. (d) MST. (e) Initial skeleton.

We test our skeleton extraction method on multiple trees from various types and with different sizes. Under most cases, the MST constructed gives a reasonable indication of how the tree's branch structure looks like (Figure 3.4).

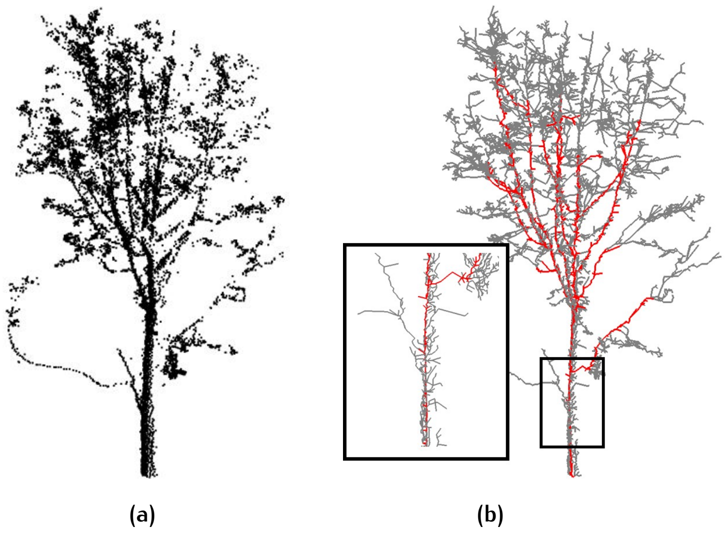


Figure 3.4: Skeleton extraction of a tall tree. (a) Tree point cloud. (b) Initial skeleton where red edges indicate the main skeletal branches.

Nevertheless, still examples exist when the pure MST cannot represent the tree skeletal structure correctly (Figure 3.5). We can observe that there are two main branches within the single trunk area, which is not correct.

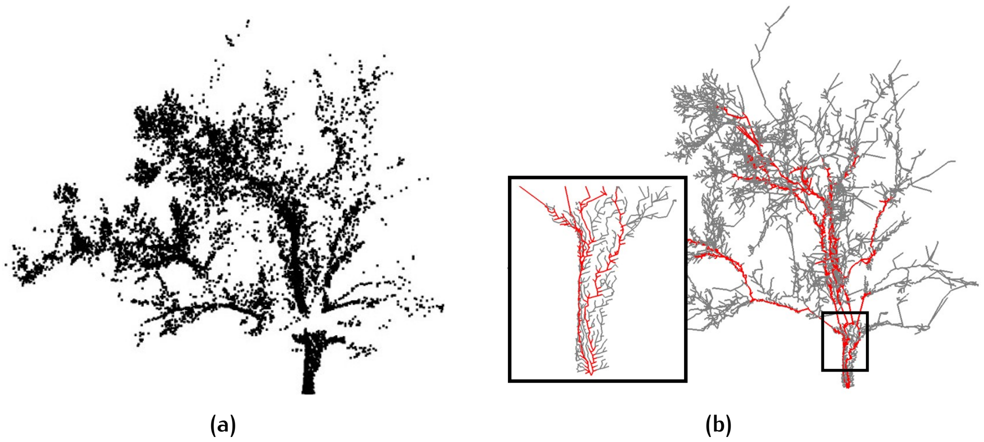


Figure 3.5: Skeleton extraction of a fat tree. (a) Tree point cloud. (b) Initial skeleton where red edges indicate the main skeletal branches..

Experimental results illustrate that trees with a short and fat shape especially don't have good skeleton extraction results. We think that the phenomenon occurs because points collected from fat plants are typically more scattered. Instead of growing compactly on a natural vertical manner, the computed MST would grow on a loose horizontal manner.

To address that problem, different possible solutions are taken into account. The first option is to manually thin the input point cloud before constructing the tree skeleton. However, the extraction result is still not satisfying, as point cloud thinning does not change the intrinsic spatial characteristic of the points.

The other option is to deliberately centralize the points to produce a more compact tree skeleton. Nevertheless, that does not apply for all the input points. Points lying within main branches should be centralized to generate condensed branches, while points near the root, the bifurcations, or the branch tips should remain to keep the original shape of the tree. Based on this idea, we begin by retrieving every single point data and identifying if it belongs to the main branches.

We can identify the main-branch points according to the tree points density change. A density map of the tree point cloud is computed and visualized in Figure 3.6. We observe that points at the tree base, the bifurcations or the branch tips typically have a sharp change in terms of their density, while points within a single branch share a relatively stable density.

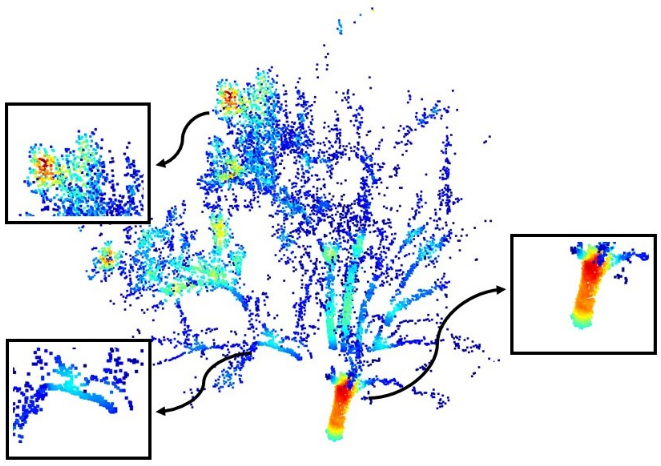


Figure 3.6: Density map of the tree point cloud. Blue color indicates low density and the red color indicates a relatively high density.

Based on the desired characteristic of the points density distribution, we identify the main-branches points by computing their density changing rate within the neighbourhood. For each point, the density changing rate $\Delta\rho$ is computed using the following equation:

$$\Delta\rho = \frac{1}{n}\sum_{i=1}^{n} \mid d_i - d \mid \tag{3.1}$$

Where n is the neighbourhood number for the specific point, d_i is the density of the i^{th} neighbouring point, d is the density of the current point. If the changing rate of density is smaller than a given threshold:

$$\Delta\rho \leq \varepsilon \tag{3.2}$$

It means that the point has a stable density within its neighbourhood. Then, we recognize the point as the main-branch point. For the main-branch point which has been identified, we apply a *mean shift* method to centralize its location. Mean shift is a simple iterative procedure, shifting each point to the average of its neighbouring [Cheng, 1995], as demonstrated in Figure 3.7.

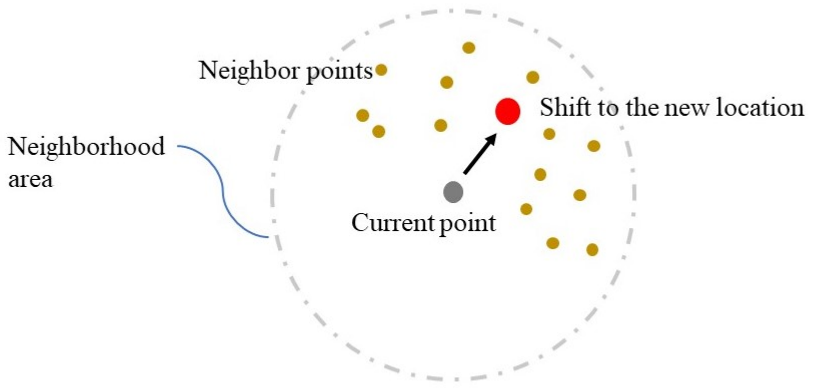


Figure 3.7: Mean shift method.

After identifying and centralizing the main-branch points, we extract the tree's skeleton over the processed tree point cloud, as shown in Figure 3.8. We can observe that now the initial skeleton extracted has a correct topological branch structure. The extracted result shows that main-branch points centralization helps to improve the topological correctness of the tree branch structure.

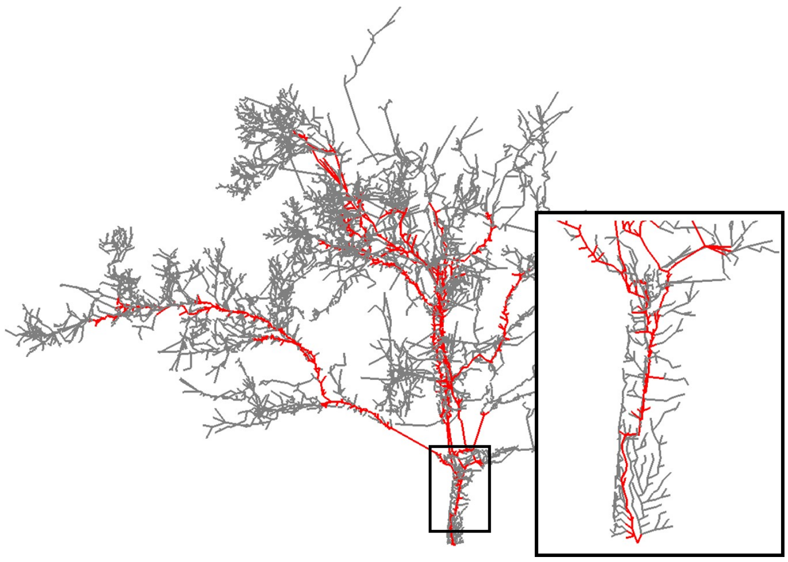


Figure 3.8: Skeleton extraction aided by the main-branch points centralization. Red edges indicate the main skeletal branches.

3.3 SKELETON SIMPLIFICATION

Having extracted the initial tree skeleton from the input point clouds, we notice that the graph has amounts of redundant vertices and edges. Most of the redundant vertices and edges don't contribute to the tree skeleton shape and thus are of little importance. Those unimportant components should be removed to simplify the tree graph. The simplification is conduct by iteratively checking the proximity between adjacent vertices. Two scenarios are taken into account during the simplification process:

- If the current vertex has one single child, then a line-simplification algorithm will be applied to check the if the current vertex can be cleared from the initial skeleton;
- If the current vertex has multiple children, then a multiple-children simplification strategy will be conducted to merge close children vertices.

The simplification process loops iteratively until the tree skeleton is refined to its simplest. The overview of the simplification process is provided in Algorithm 3.1.

Algorithm 3.1: Skeleton Simplification

```
   Input: Initial skeleton of the tree
   Output: The simplified tree skeleton
1  $\delta \leftarrow True$;
2  while $\delta$ do
3      $\delta \leftarrow False$;
4      for V in skeleton do
5          if V has single child then
6              conduct single-child simplification;
7              $\delta \leftarrow True$;
8          else
9              conduct multi-children simplification;
10             $\delta \leftarrow True$;
11 return simplified skeleton
```

3.3.1 Assigning confidences to vertices and edges

We assign weights to vertices and edges in the initial tree skeleton to further guide the simplification process. We want to keep important vertices and main branch edges while ignoring short branches and noisy points. A number of previous works [Guo et al., 2018] suggest utilizing the point density, together with the point orientation vector extracted from Principal Component Analysis (PCA), to indicate the importance of the vertex. However, the weights evaluated in such way are significantly dependent on the quality of the scanned points and thus become unreliable when encountered with poor scanning issues.

Instead of weighting from the density, we weight each vertex according to its length of subtree, which is computed as the sum of the length of all edges within the subtree of the vertex. Through such way, high-connective vertices close to the tree base area get heavier weights while low-connective vertices near the tree crown get smaller weights. One advantage is that the weighting process is not sensitive to input points density, which makes it robust to data with different scanning qualities.

Accordingly, each branch edge is weighted as the average of the subtree length of its two ending vertices:

$$w_e = \frac{L_{v_1} + L_{v_2}}{2} \tag{3.3}$$

Where w_e is the weight of the edge, L_v is the subtree length of the specific vertex.

Figure 3.9 shows the weighs assigned to vertices and edges, which is also briefly abstracted in Figure 3.10. It is noticed vertices and edges on the tree crown have consistent low weights, while near the tree base area small branches have drastically small weights compared to the main tree branches. Such characteristic helps us clear away noisy branches at the trunk, and, at the same time, keep the small leave twigs at the crown.

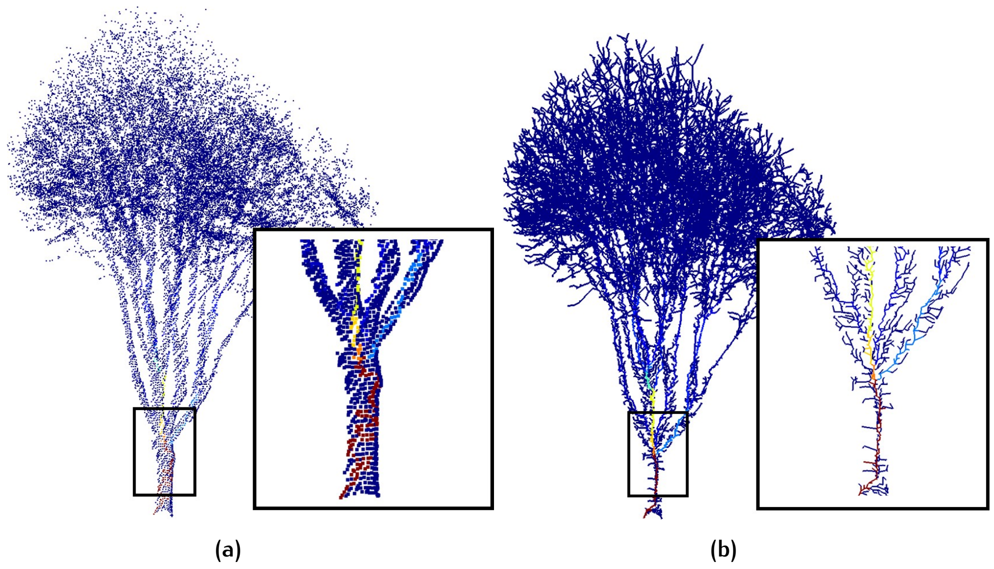


Figure 3.9: Weights assigned to the vertices and the edges. Red indicates high weight while blue indicates low weight. (a) Vertices weights. (b) Edges weights.

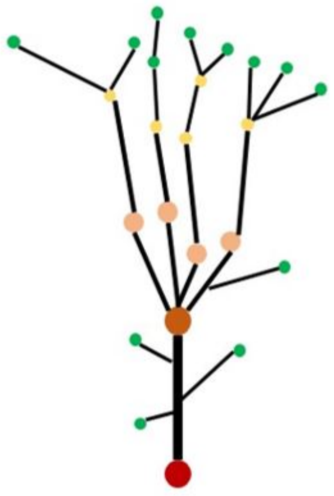


Figure 3.10: An illustration of the vertex and edge weights. Red and thick lines indicate high weights, green and thin lines indicate low weights.

To clear away the noisy small branches at the tree base, we retrieve all the vertices in the initial tree skeleton. For each vertex, we compute the weight ratio between it and its parent vertex:

$$\delta w_i = \frac{w_i}{w_p} \tag{3.4}$$

Where w_i is the weight of the i^{th} vertex and w_p is the weight of the parent vertex. If δw_i is smaller than a given threshold:

$$\delta w_i \leq \tau \tag{3.5}$$

We assume that the current vertex is a noisy data point, which is culled away together with its subtree branches. Figure 3.11 shows that after removing the noisy vertices and small branches, we obtain a more concise tree skeleton.

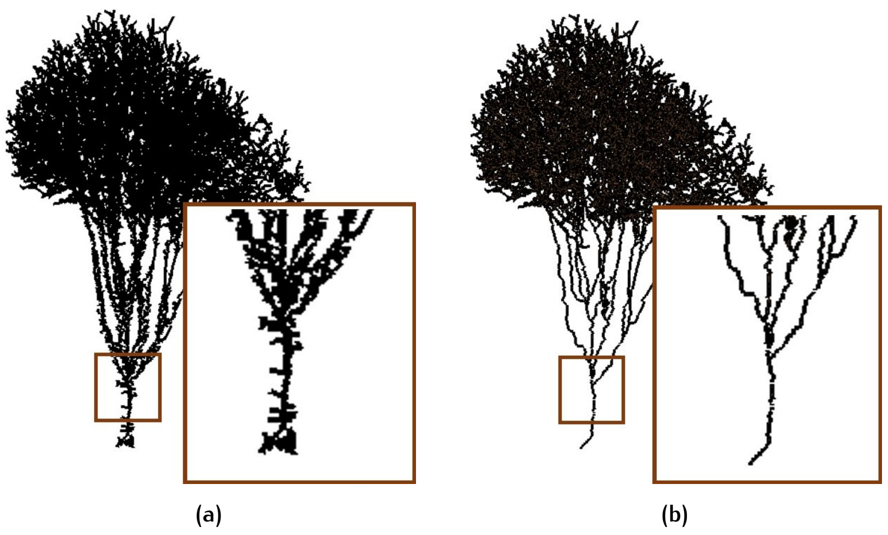


Figure 3.11: Eliminating small noisy branches. (a) Initial tree skeleton. (b) Skeleton after removing noisy edges.

3.3.2 Simplifying the vertex with one single child

For vertex which has only one single child, we specify the problem as a line simplification problem. According to [Wu and Marquez, 2003], the Douglas-Peucker algorithm is regarded as the most effective line simplification algorithm in literatures. Douglas-Peucker is applied to simplify line segments by gradually decimating those unimportant points on the segments. The more proxy the current point is to the segment, the less important it is. Hereby, we use a similarity indicator α to describe the closeness between the point and the line segment:

$$\alpha = \frac{d}{r} \tag{3.6}$$

Where d is the distance between the current point and the line segment formed by its parent and its child, r is the distance threshold of an edge in the tree skeleton which controls the simplification process. As illustrated in Figure 3.12, if the indicator value α is smaller than a given threshold:

$$\alpha \leq \sigma \tag{3.7}$$

We assume that the current point is unimportant and therefore clear it from the graph.

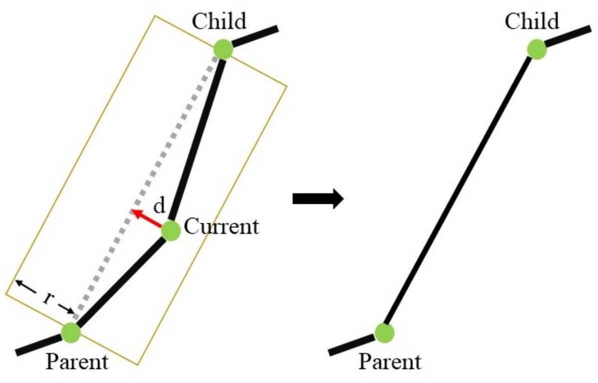


Figure 3.12: Single-child vertex simplification.

3.3.3 Simplifying the vertex with multiple children

For vertex which has multiple children vertices, we check the closeness between its children vertices. If the children are too close with each other, then we merge them into a new child vertex. To evaluate the proximity between a specific pair of children vertices, we propose the bidirectional similarity indicator, which is defined as follow:

$$\alpha_{1\leftrightarrow 2} = \min\left(\frac{l_1 \sin\theta}{r_2}, \frac{l_2 \sin\theta}{r_1}\right) \tag{3.8}$$

Where $\alpha_{1\leftrightarrow 2}$ indicates the closeness between the pair of two children vertices, l represents the length of the edge between one specific child vertex and its parent, θ is the angle between two edges, and r is the distance threshold of an edge (see Figure 3.13).

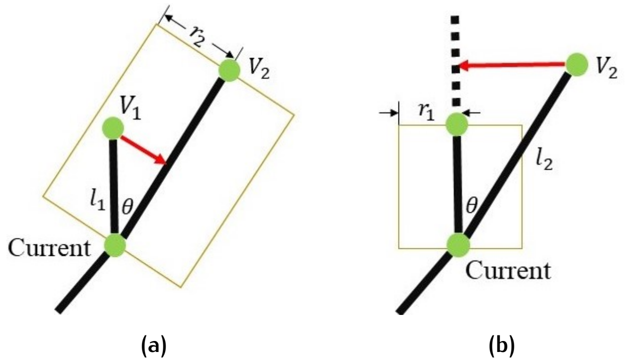


Figure 3.13: Bidirectional similarity indicator. (a) Indicator computed from V_1 to V_2. (b) Indicator computed from V_2 to V_1.

The reason we name the similarity indicator as *bidirectional* is that the indicator computed from different direction (i.e. from V_1 to V_2 or from V_2 to V_1) will have different values. As the example in Figure 3.13 illustrates, the indicator value computed aligning V_1 to V_2 will have a smaller value compared with the other way around. Therefore, we select the minimum indicator value to evaluate the proximity between a pair of two children vertices. The smaller the indicator value, the more possible it will be to merge vertices together.

Generally, one bifurcation vertex has more than two children vertices. Therefore, for each bifurcation vertex, we retrieve every possible pair of its children and compute the corresponding bidirectional similarity indicator. We identify the pair with the smallest indicator value $\alpha_{1\leftrightarrow 2}$. If the identified indicator is smaller than a given threshold:

$$\alpha_{1\leftrightarrow 2} \leq \sigma \tag{3.9}$$

Then we merge the pair of vertices into a new vertex. The merged new vertex location p_{new} is computed as the weighted average of the two old vertices, where, as declared in Section 3.3.1, the weight of each vertex is represented as its subtree length:

$$p_{new} = \frac{p_1 w_1 + p_2 w_2}{w_1 + w_2} \tag{3.10}$$

Some special cases exist when a small indicator value doesn't ensure high similarity between two edges. As depicted in Figure 3.14, an extremely short edge will result in a small indicator value but doesn't necessarily mean that it is similar to the other edge. Merging vertices under such circumstances will cause unreasonable orientations of branches, and therefore should be avoided during the simplification process.

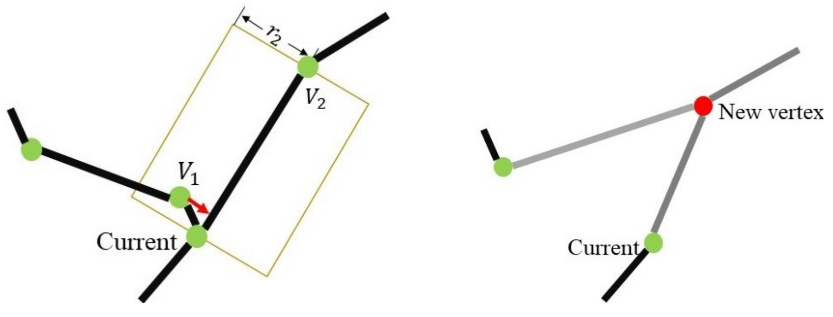


Figure 3.14: Unreasonable merging. (a) An illustration how the indicator is computed. (b) Merging result where grey edges are new-produced branches.

To avoid unreasonable merging, we forbid clustering edges together if they differ a lot in the length (i.e., the long edge should not exceed the 2 times length of the short edge). Figure 3.15 shows how our strategy helps to maintain the natural topological structure of the tree branches during the merging process. It is observed that the simplification result after we forbid short-edges merging has more natural and smoother growth direction of the tree branches.

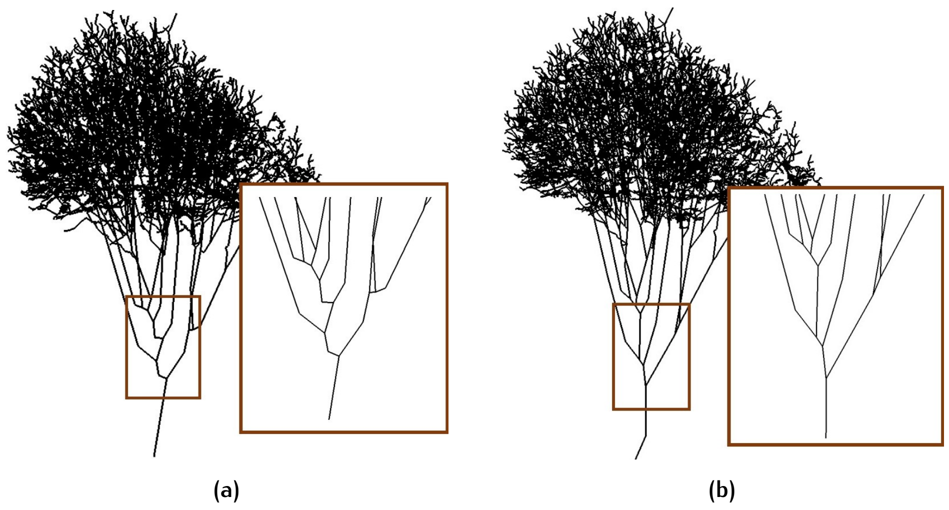


Figure 3.15: Comparison of different simplification method. (a) Normal simplification. (b) Simplification where the short-edges merging is forbidden.

The initial tree skeleton is iteratively retrieved, with its proxy vertices recognized and clustered, until no further vertices can be merged. Figure 3.16 shows the simplified tree skeleton in comparison with the initial tree skeleton.

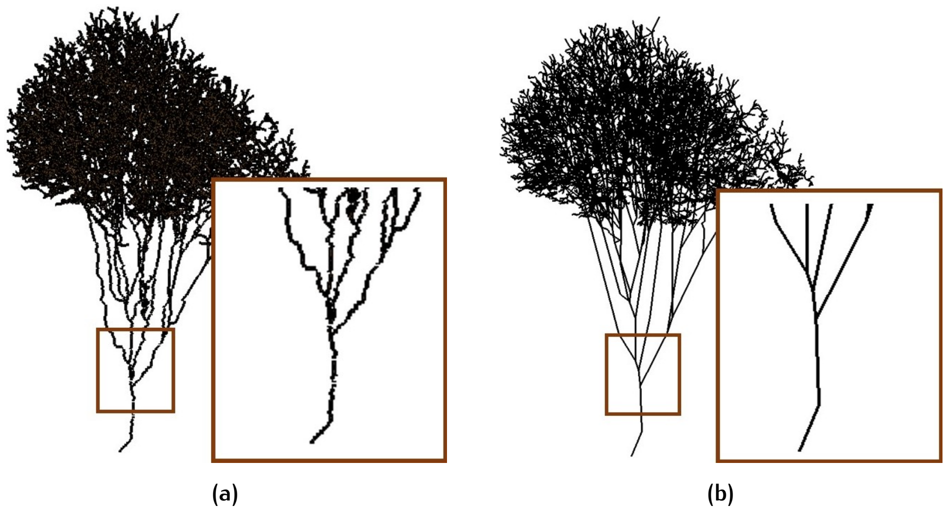


Figure 3.16: Skeleton simplification. (a) Initial skeleton without noisy branches. (b) Simplified skeleton.

3.4 BRANCH FITTING

The next step after skeleton simplification is inflating the tree branch geometry based on the tree skeleton. To precisely model the geometry of tree branches, [Guo et al., 2018] utilized a B-spline curve fitting method, [Chen et al., 2008] applied a sketch-based modelling system which reconstructs tree branches from user's free-hand drawing.

Apart from existing methods, we want to explore a simple way to reconstruct the geometry of tree branches while avoiding users' interactions as much as possible. To achieve that, we explore a cylinder-fitting approach. It is observed that generally trees branches in a natural or urban environment can be well represented as cylinders (Figure 3.17). According to [Markku et al., 2015], the cylinder is the most robust primitive in terms representing the geometry of the tree branches, even with holes and noises in the dataset. Moreover, compared with complex curve-fitting, cylinder-fitting method is relatively easy and fast in computation.

(a)

(b)

Figure 3.17: Tree branches in the nature. (a) Tree branches with regular growth. (b) Tree branches with irregular growth.

We want the reconstructed branch cylinders to accurately fit the input points. Therefore, we exploit an optimization-based approach to obtain accurate branch geometry. For those small tree branches which don't have sufficient supportive points or suffer from missing scan data, we apply an allometric tree growth theory to derive their geometries.

3.4.1 Main trunk fitting

The main trunk close to the tree base area is often the branch with the highest density of supportive points. It is important to accurately model the geometry of the trunk, as it lays the foundation for the propagation of the rest subsequent branches. We begin by identifying the points which belong to the main trunk. We build the *kd*-tree over the point cloud and assign all the points to the corresponding branch edges through the *kd*-line intersection query. Figure 3.18 shows how the points are segmented into different tree branch parts.

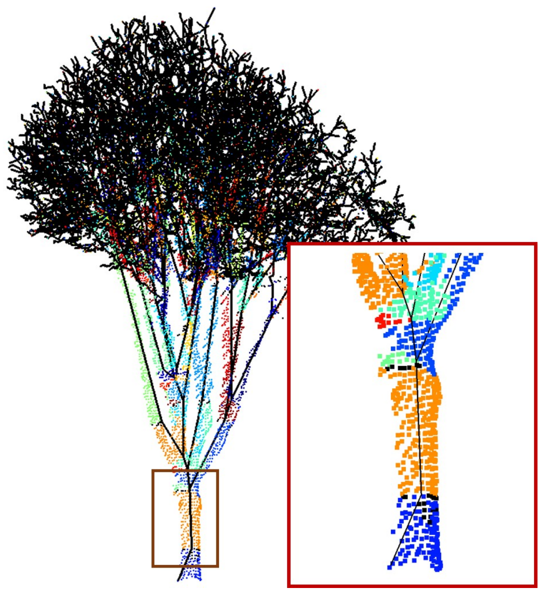


Figure 3.18: Points assigned to different branch edges. Different colours indicate different tree branch parts.

The next step is to fit a cylinder to approximate the branch geometry on the basis of the corresponding trunk points. This is a typical non-linear least squares problem. We hereby define our input data, parameters to be solved, and the objective function as follow (Figure 3.19):

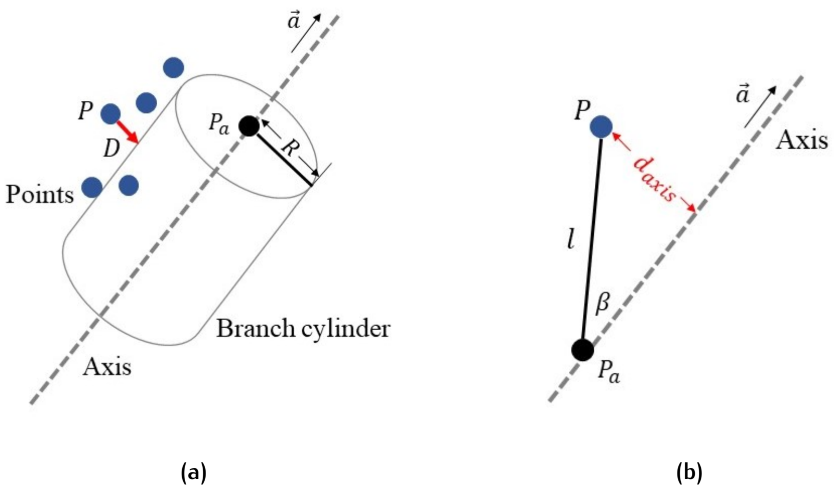


Figure 3.19: Cylinder fitting. (a) Parameters to be solved. (b) Distance computation from the point to the cylinder axis.

- **Input data** are the position P of the input points;
- **Parameters to be solved** are the axis direction vector $\vec{a}$ of the cylinder, the position P_a of the end point on the axis, and the radius R of the cylinder;
- **Objective function** is defined as the sum of the squared distance $\min \sum D$ from the points to the branch cylinder.

To obtain the distance D between the point and the cylinder, we first obtain the distance between the point and the axis d_{axis} and then subtract it from the cylinder radius R.

$$D = | d_{axis} - R | \tag{3.11}$$

While d_{axis} can be computed as follow:

$$d_{axis} = l \sin \beta = \|P - P_a\| \sin \left(\arccos \frac{\vec{a}(P - P_a)}{\|P - P_a\|} \right) \tag{3.12}$$

As delineated in Figure 3.19, l is the distance between the current tree point P to P_a, which is the end point on the cylinder axis, β is the angle formed by the axis and the vector $\overrightarrow{P_a P}$. It is denoted that the axis direction vector $\vec{a}$ is supposed to be normalized beforehand.

Each tree point defines a distance expression:

$$f(p) = D = | d_{axis} - R | \tag{3.13}$$

And our objective is to minimize the sum of f(p) for all the tree-points:

$$\min \sum_{i=1}^{n} f(p)_i \tag{3.14}$$

For solving the 7 unknown parameters, we have multiple expressions as many as the number of the trunk points. That introduces huge redundancy and thus enables an accurate solution. Levenberg Marquardt algorithm to is applied to solve the non-linear least squares problem [Moré, 1978].

To further improve the solution quality, we repeat the non-linear least square process and introduce the weights for each point during the second computation iteration. We want to give heavy influence to the points closer to the cylinder and relatively low influence to the points which are far from the cylinder. Hence, weights are assigned according to the point's distance to the cylinder. The weight for one specific point is defined as follow:

$$w_i = 1 - \frac{D_i}{D_{max}} \tag{3.15}$$

Where D_i represents the distance between the current i_{th} point and the cylinder obtained from the initial adjusting process, and D_{max} is the maximum distance among all the points to the cylinder. Through such way, we normalize all the weights of the points to the range between 0 and 1.

The objective function is denoted accordingly:

$$\min \sum_{i=1}^{n} f(p)_i w_i \tag{3.16}$$

After the cylinder-fitting process, we obtain the accurate geometry of the tree trunk, as depicted in Figure 3.20.

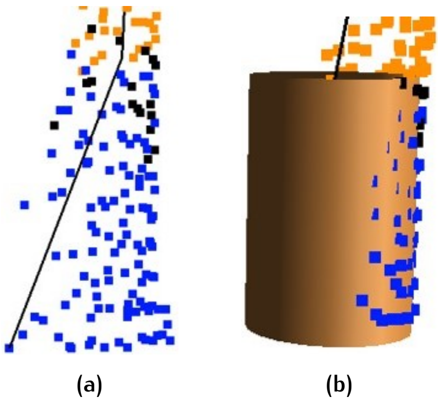


Figure 3.20: Trunk cylinder fitting. (a) Trunk points. (b) Fitted cylinder.

3.4.2 Small branch propagation

Since the point cloud usually becomes noisier and contains more outliers when it comes to the tree crown and branch tips, fitting an accurate cylinder to a small branch will be almost infeasible. Instead, we apply an allometric rule to obtain plausible estimates for the rest of the tree branches [Guo et al., 2018]. The radius of a branch edge is proportional to its weight, which, as defined in Section 3.3.1, is the average of the subtree length of its two ending vertices. We compute the radius of the rest branch edges using the following equation:

$$R_{e_i} = R_t(\frac{w_i}{w_t})^{1.1} \tag{3.17}$$

Where R_{e_i} is the radius of the i^{th} branch edge, R_t is the radius of the trunk which is obtained from Section 3.4.1, w is the weight of the specific branch edge, parameter 1.1 is used to control the branches propagating speed. Figure 3.21 shows the derived tree branches model from the constructed tree skeleton.

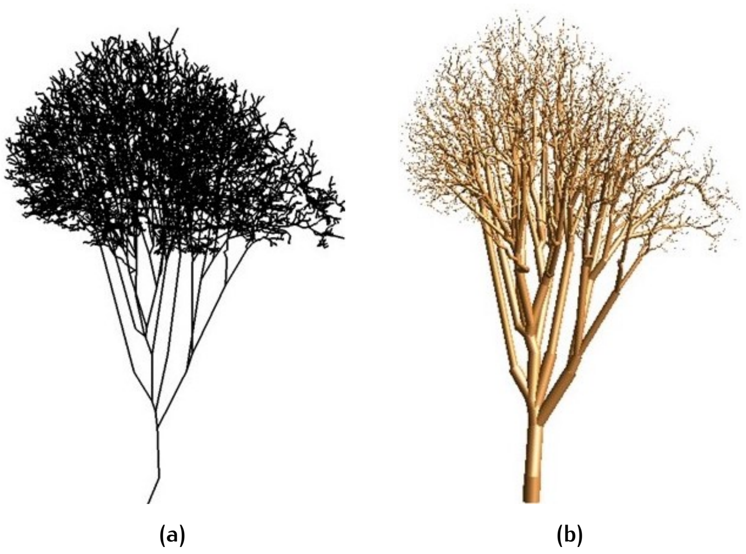


Figure 3.21: Branch fitting. (a) Tree skeleton. (b) Tree branch model.

3.5 ADDING REALISM

To further add realism, we add leaves and texture to the reconstructed tree models. Since it's almost impossible for us to capture the geometry and texture characteristics of leaves from laser scanned data, we are unable to reconstruct accurate leaves purely from the input point cloud. Therefore, leaves reconstruction is out of the research scope. However, we would like to render leaves to convey the realism of the model. We randomly generate oriented leaves at the end of each branch. The leaves are rendered as textured quads in the rendering. Figure 3.22 shows the final reconstruction.

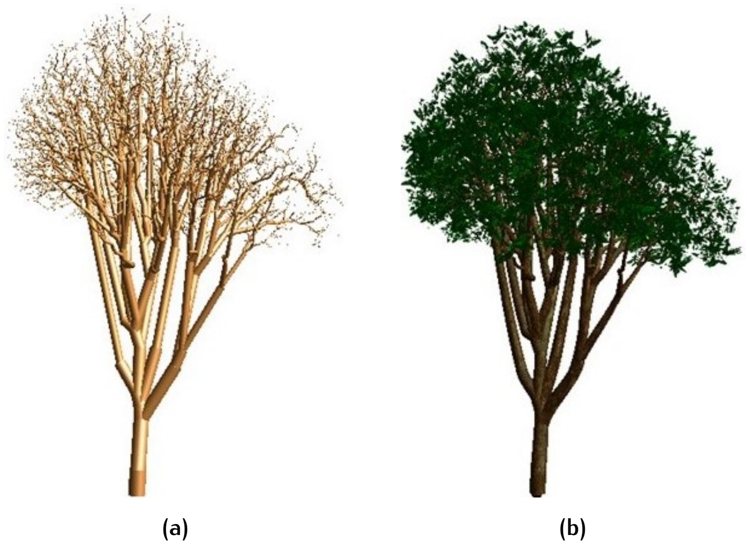


Figure 3.22: Adding leaves and texture. (a) Tree branch model. (b) Final tree model.

4 RESULTS AND DISCUSSION

This chapter aims to provide the experiment results for the study. First, a detailed description of the implementation and prototype is provided in Section 4.1. Then, to demonstrate the robustness of our method, we test our algorithm over a wide range of tree point clouds with various species, sizes, and data sources (Section 4.2). Finally, some discussions considering parameters, comparisons with previous works, our limitations and future applications are made in Section 4.3.

4.1 IMPLEMENTATION DETAILS

4.1.1 Datasets

To develop and test the proposed tree reconstruction method, several point cloud datasets have been collected. These test datasets contain point clouds from publicly available point cloud repositories, the Floriade Project of Almere, and the AHN dataset. These point clouds include various tree species and types. Also, a wide range of data sources is covered, i.e., static laser scans, mobile laser scans, and airborne laser scans.

4.1.2 Software prototype

We use C++ as the programming language to implement the algorithms described in chapter 3. The development environment is Microsoft Visual Studio 2017. We select C++ because it enables high computation efficiency. Also, many necessary libraries for the tree skeleton graph modelling and the fine model rendering are provided in C++, which are listed as follow:

- **BGL**: Boost Graphic Library; it provides support for graph structures, among which the adjacent list is used to construct graphs and the shortest path is used to compute the MST; Furthermore, BGL allows users to define the needed attributes for the vertices and edges.

- **OpenGL**: Open Graphics Library; it is used to render 2D and 3D vector graphics. It achieves hardware-accelerated rendering performance by directly interacting with the GPU;

- **Qt Library**: It is mainly used for the User Interface design and creation of the tree modelling application platform.

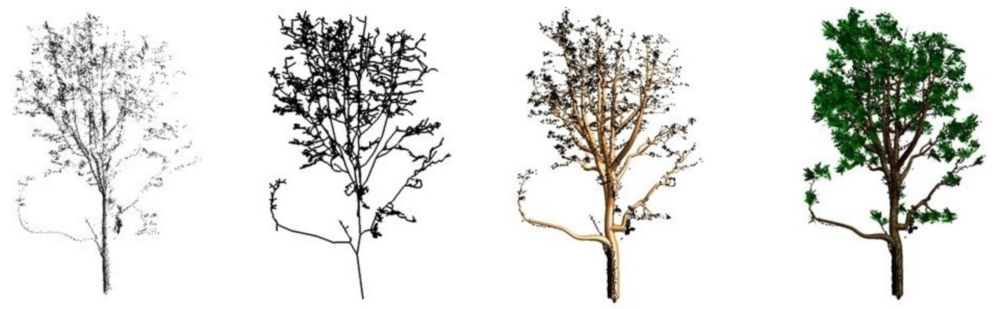
- **Easy3D**: It is a C++ library for processing and rendering 3D data, which is developed by Nan [2019]. In this thesis we mainly use Easy3D for efficiently rendering input tree point clouds and reconstructed tree models.

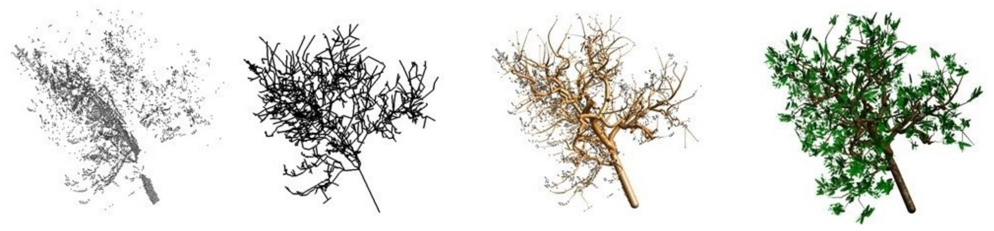
Besides the C++ libraries, tools like Mapple, which is a tool for visualizing and editing 3D point clouds [Nan, 2018], will be used for the visualization and segmentation individual tree point clouds.

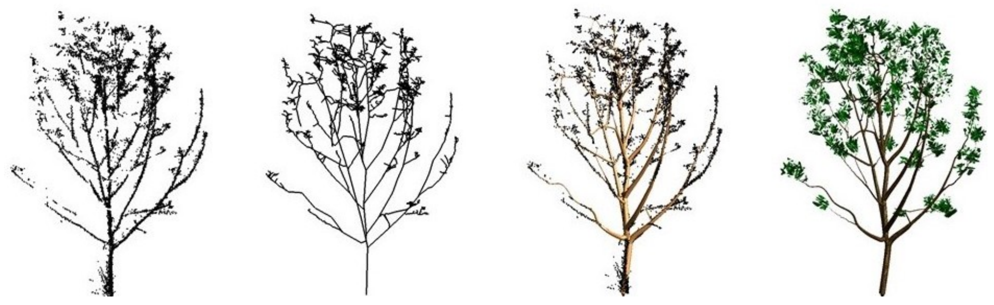
4.2 RESULTS

To verify the effectiveness and capacity of our proposed methodology, we reconstruct a variety of trees with different species and branch structure. Among them, some trees are tall and slim, while others are short and fat. Most trees can be validly reconstructed, as delineated in Figure 4.1.

Our reconstruction results demonstrate that our proposed methodology can process various type of trees with different sizes. Especially, we show a tree with missing data due to occlusion (Figure 4.1b). The algorithm is able to complete the missing region and reconstruct plausible branch structure.


(a)


(b)


(c)

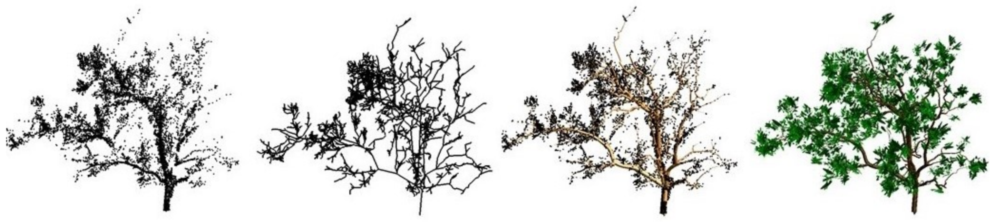
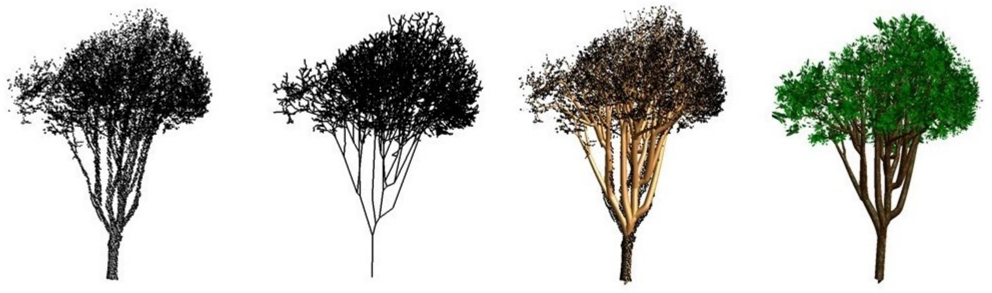
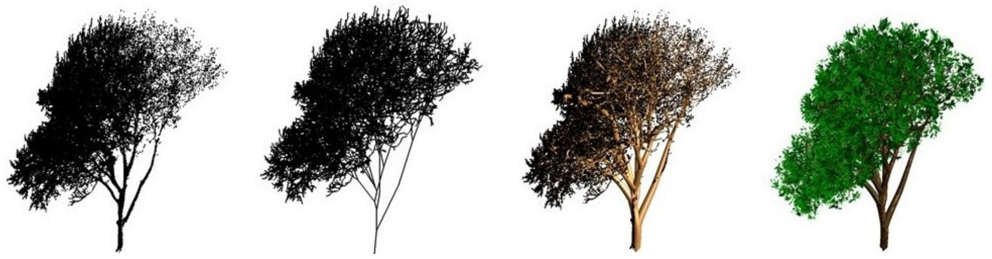

(d)

Figure 4.1: Reconstructed models for various trees. From left to right: point cloud; skeleton; tree branches; tree final model.

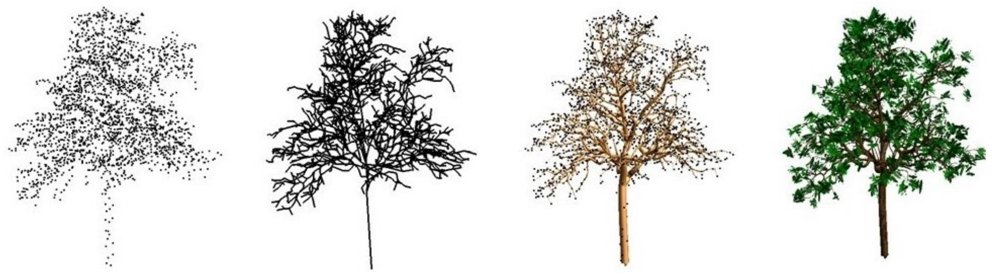
In addition, we also test our methodology over tree scans from various data sources, including static scanning, mobile scanning as well as airborne scanning. The modelling results are shown in Figure 4.2.



(a)



(b)



(c)

Figure 4.2: Reconstructed models from various data sources. (a) Mobile scanning. (b) Static scanning. (c) Airborne scanning.

It is observed that point clouds collected by mobile scanning or static scanning always have high quality, and thus will be accurately reconstructed. On the other hand, trees sampled by airborne scanning are poorly collected and are difficult to reconstruct. Since our approach is mainly data-driven, insufficient input points will compromise the accuracy of the modelling results. Nevertheless, our approach is able to keep the main structure of the tree branches. The last example of Figure 4.2 shows an individual tree point cloud from ANH3 dataset, which is poorly scanned and contains sparse points. Still, our approach can reconstruct the basic topological structure of the tree branches.

We evaluate the geometrical accuracy of the modelling results by computing the mean distance between the input points and the generated tree branch model. The reconstructed model is re-sampled as vertex sets, where for each vertex we find its closest point from the input laser scanned data. Then we record the distance between the vertex and the targeted point. Tree No.1 (Figure 4.1a), Tree No.2 (Figure 4.1b) and Tree No.3 (Figure 4.1d) are randomly chosen for this evaluation. Results in Table 4.1 suggest that our approach can generate tree models which fit closely to the input point cloud data and thus ensures high geometrical accuracy.

	Tree No.1	Tree No.2	Tree No.3
Height	5.61m	10.05m	10.61m
Mean Distance	2.76cm	10.04cm	6.25cm
Standard Deviation	0.02	0.08	0.06

Table 4.1: Geometrical accuracy evaluation

Figure 4.3 shows the error distance computed from each point to the tree branch model, taking the tree of Figure 4.1a as an example. Blue colour indicates a low value and red colour indicates a high value. It is denoted that points lying within the main branches typically fit closely to the model, while points near the branch tips or branch turning angles have high values of error.

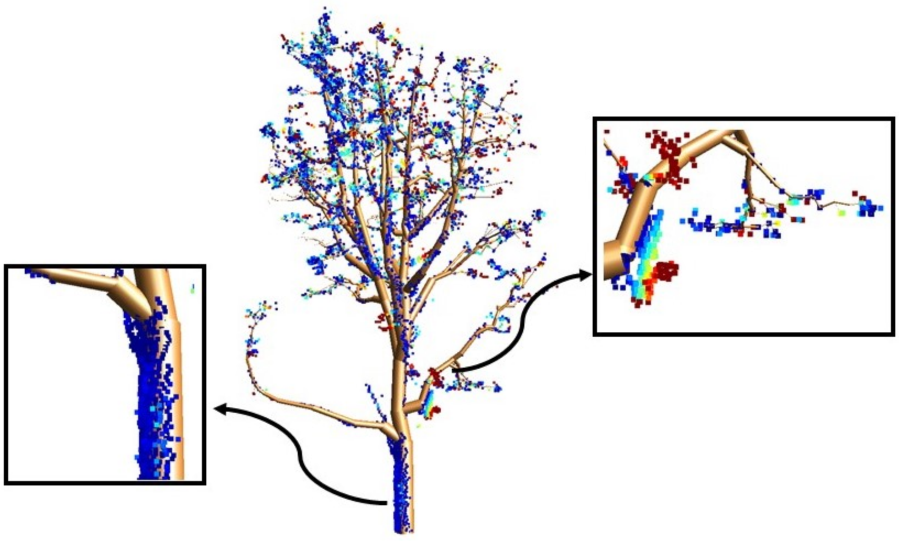


Figure 4.3: Error visualization from points to the model.

4.3 DISCUSSION

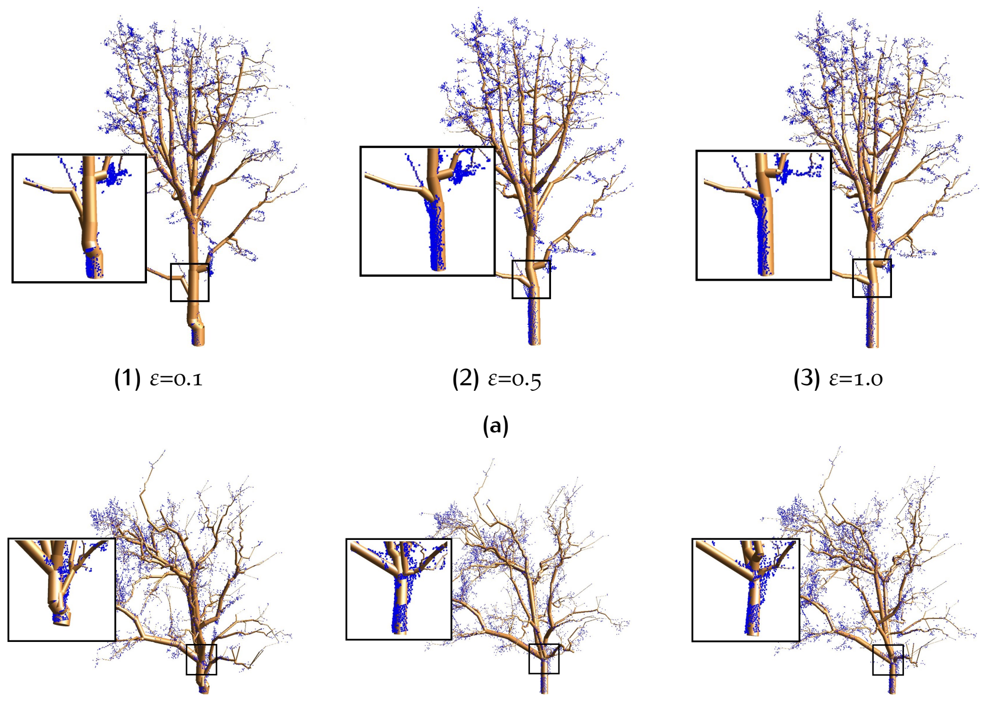
4.3.1 Parameters

As described in Chapter 3, a number of parameters are introduced to control the tree modelling process, including the threshold ε of the density changing rate, the threshold τ for eliminating noisy branches, and the threshold σ for simplifying the skeleton. This section discusses how different parameters influence the modelling results, based on which, we choose the threshold values that best fit our methodology.

Threshold of the density changing rate

The threshold ε of the density changing rate is introduced during the main-branch points centralizing process (see Section 3.2). A very small ε means that nearly no points will be clustered or centralized, while a very large ε indicates that points will be over clustered and thus cause the tree skeleton to lose its original shape. We test the value of ε from 0.1 to 1 and the results are shown in Figure 4.4.

It is observed that when ε is extremely small, main-branch points can be hardly recognized and thus centralized, resulting in unnatural trunk structures. On the other hand, an extremely big ε value will over-centralize points. According to our trial and error testing results, we select 0.5 as the value for threshold ε.



(1) ε=0.1 (2) ε=0.5 (3) ε=1.0

(a)

(1) ε=0.1 (2) ε=0.5 (3) ε=1.0

(b)

Figure 4.4: Reconstruction results using different ε. (a) Tree No.1. (b) Tree No.2.

Threshold for eliminating noisy branches

The threshold τ for eliminating small noisy branches is introduced when we assign weights to the vertices in the tree skeleton and remove small noisy branches near the tree base area (see Section 3.3.1). If the subtree length ratio between one specific vertex and its parent is below τ, then the vertex together with its subtree is cleared away. A small τ means that many small noisy branches which should be removed will be retained, while a very large τ indicates that some normal branches will also be culled away. We test the value of τ from 0.005 to 0.04 and the results are shown in Figure 4.5.

As illustrated in Figure 4.5, when τ is too small, many noisy branches near the tree base area can't be removed; when τ is too large, there is also a possibility that long main branches will be wrongly culled away. Therefore, it is important to select an appropriate value for threshold τ. Based on our experiments over different tree point clouds, we choose 0.02 as the threshold value.

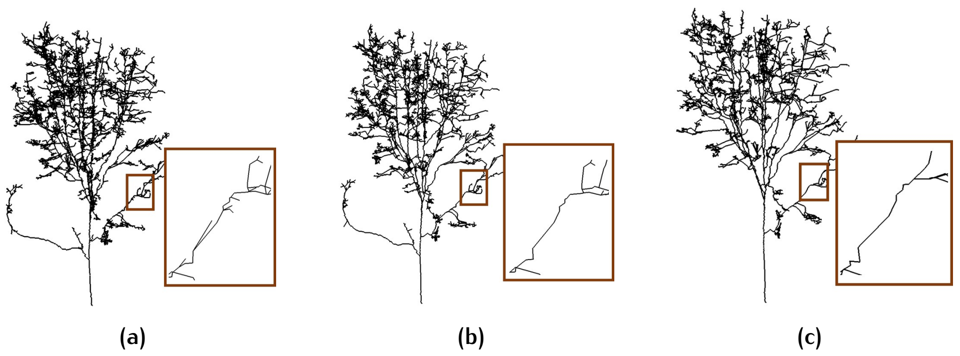


Figure 4.5: Noisy branch eliminating results using different τ values. (a) $\tau = 0.005$. (b) $\tau = 0.02$. (c) $\tau = 0.04$.

Threshold for simplifying skeleton

The simplification threshold σ is introduced during the tree skeleton simplification process (see Section 3.3.2, Section 3.3.3), where we utilize an indicator α to evaluate the proximity between the adjacent vertices.

$$\alpha = \frac{d}{r} \tag{4.1}$$

The equation above defines how the indicator α is generally computed. d is the distance between one vertex and the corresponding branch edge of its adjacent vertex. r is the radial distance threshold for that edge. To compute r for each branch edge, we first select trunk points which are below 2% of the tree height above the ground. By projecting the trunk points onto the ground and applying a circle fitting, we obtain an initial radial distance threshold r_t for the trunk edge. Then an allometric rule is utilized to compute the distance threshold of each edge:

$$r_{e_i} = r_t(\frac{w_i}{w_t})^{1.1} \tag{4.2}$$

Where w is the weight of the specific edge (see Section 3.3.1). The equation above illustrates that the indicator α determines the relative proximity between adjacent vertices. Accordingly, a very small threshold σ for the indicator will make it tough for close vertices to merge together, while a very big σ will cause oversimplification. We test the value of σ from 0.5 to 3 and the results are shown in Figure 4.6.

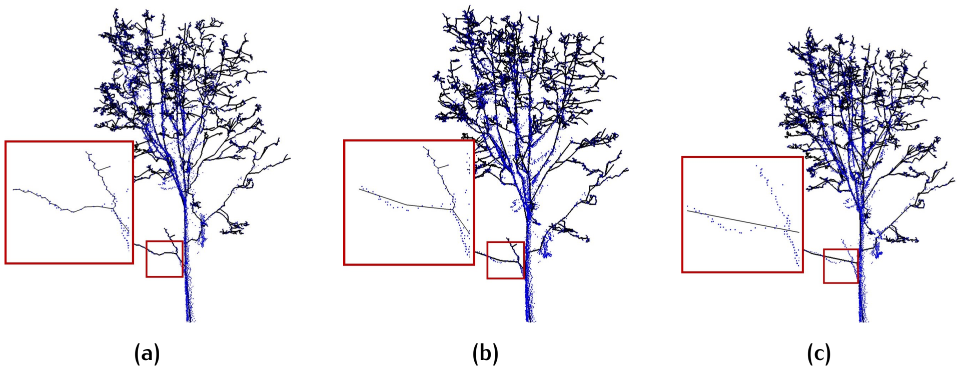
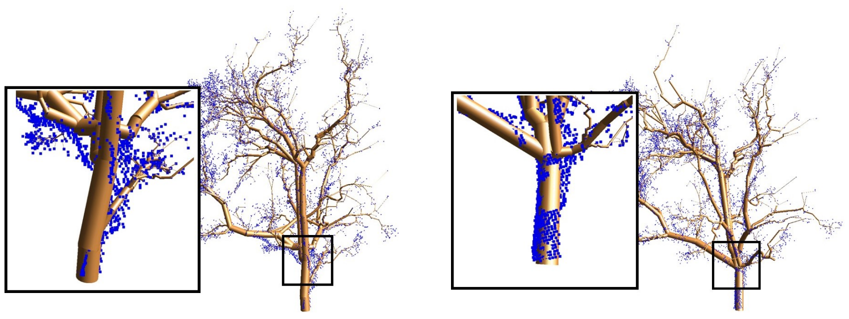


Figure 4.6: Simplification results using different σ values. (a) $\sigma = 0.5$. (b) $\sigma = 1.5$. (c) $\sigma = 3$.

We can draw a conclusion that when σ is too small, many redundant vertices and edges cannot be simplified; when σ is too large, the tree skeleton will be oversimplified that it doesn't keep the natural shape anymore. Based on our experiments, we choose 1.5 as the threshold value.

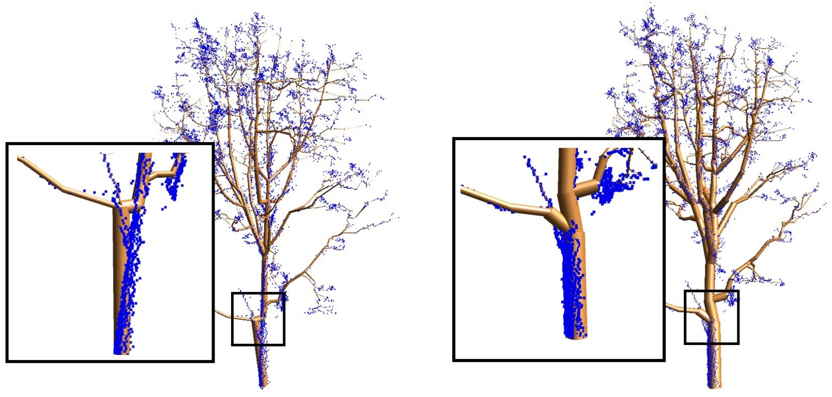
4.3.2 Comparison

We compare our modelling results to Livny's work (2010) as it's the closest related to our work. Given the same point cloud, our algorithm is capable of reconstructing tree models with higher topological accuracy and geometrical accuracy. That results from several reasons: First, we identify and centralize main-branch points, which will in return generate more plausible tree skeleton; Second, we avoid unreasonable merging that may lead to abnormal branch growing angles; Last, we make use of the positions of input points to fit the branch geometry. We make a visual comparison between our reconstruction approach and Livny's approach, which is shown in Figure 4.7.



(1) Livny's reconstruction (2) Our reconstruction

(a)



(1) Livny's reconstruction (2) Our reconstruction

(b)

Figure 4.7: Comparison between Livny's method and our method. Our reconstruction demonstrates both topological accuracy (a) and geometrical accuracy (b).

4.3.3 Limitations

Our proposed approach can successfully reconstruct accurate and detailed 3D tree models from point clouds. However, it still has some limitations. First, our approach is mainly data-driven. For those input point clouds with high data quality, it is able to accurately reconstruct the tree branches; However, for those poorly scanned data with sparse points, while our method can reconstruct a plausible topological structure of the tree branches, it is unable to achieve sufficient geometrical accuracy; Also, our work does not consider natural growing rules for tree branches (i.e. branch split angle, branch growing length, etc.). The combination of our data-based approach and the rule-based approach will be a promising field for our future study.

4.3.4 Applications

Our proposed approach enables accurate reconstruction of 3D trees from point clouds. The generated tree models can be employed in many applications. Most primarily, it opens up the chance for automatically obtaining precise tree attributes (for instance, the tree height, the trunk thickness, and the Diameter at Breast Height). That will save a lot of time and labouring efforts compared to the traditional manual measuring method. Implicit tree variables can also be derived from the tree models, i.e. by computing the sum volume of all reconstructed branch cylinders we are able to estimate the wood volume from the tree, and further quantify biomass and other forest-related variables. Besides, accurate tree models with synthesized leaves and textures can be employed in urban landscape design and visualization, to enhance the realism of the digital scenes. More details are provided in Appendix A.

5 CONCLUSION AND FUTURE WORK

In this thesis study, we propose an automatic approach to accurately reconstruct 3D tree models from individual tree point clouds. Generally, the reconstruction results are promising.

During the modelling process, both the geometrical accuracy and topological fidelity of the tree are taken into consideration. One novelty of our work is that we aid the skeleton construction process with the main-branch point centralization and further abandon unreasonable skeleton simplification, which comtributes to improve the quality of the generated tree branch structure. Moreover, an optimization-based approach is employed to accurately reconstruct the geometry of the tree branches.

Tested results in Chapter 4 reveal that our method is robust in dealing with various types and sizes of the trees. As long as the input point cloud maintains significant branch structure, our method is able to generate tree models with good quality. When it comes to the airborne laser scanned data with sparse points, our method can reconstruct plausible a topological structure of tree branches. Nevertheless, the geometrical accuracy may be compromised.

In future we would like to continue in the following directions:

- We would like to conduct pre-segmentation for the input point clouds containing multiple trees. As our method only works for an individual tree point cloud now, automatic segmentation will expand our algorithm to a broader range of applications;
- As there are many irregular shapes of tree branches in nature, we will further consider fitting free-form surfaces instead of cylinders to model the branches geometry more precisely;
- Images of trees can act as a supplement data source for point clouds, from where we are able to achieve higher modelling quality. For example, from images we can extract detailed tree textures to enhance better realism;
- For now we use uniform parameters for all input point clouds. As different trees typically have different branch growing pattern, computing specific parameters from the specific individual point cloud may achieve higher modelling quality. We will consider involving that into our future work;
- Currently our approach is mainly data-driven. In future we would like to combine our approach with rule-based approaches. By applying botanical knowledge of the tree growth, for instance the branch rolling angle and the growing unit, we want to generate tree models with more plausible branches structure so as to achieve higher topological fidelity.

BIBLIOGRAPHY

Brandtberg, T., Warner, T. A., Landenberger, R. E., and McGraw, J. B. (2003). Detection and analysis of individual leaf-off tree crowns in small footprint, high sampling density lidar data from the eastern deciduous forest in north america. *Remote sensing of Environment*, 85(3):290–303.

Bucksch, A., Lindenbergh, R. C., and Menenti, M. (2009). Skeltre-fast skeletonisation for imperfect point cloud data of botanic trees. Eurographics.

Calders, K., Origo, N., Burt, A., Disney, M., Nightingale, J., Raumonen, P., Åkerblom, M., Malhi, Y., and Lewis, P. (2018). Realistic forest stand reconstruction from terrestrial lidar for radiative transfer modelling. *Remote Sensing*, 10(6):933.

Chen, X., Neubert, B., Xu, Y.-Q., Deussen, O., and Kang, S. B. (2008). *Sketch-based tree modeling using Markov random field*, volume 27. ACM.

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799.

Cornea, N. D., Silver, D., and Min, P. (2007). Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization & Computer Graphics*, (3):530–548.

Deussen, O., Hanrahan, P., Lintermann, B., Měch, R., Pharr, M., and Prusinkiewicz, P. (1998). Realistic modeling and rendering of plant ecosystems. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 275–286. ACM.

Dey, T. K. and Sun, J. (2006). Defining and computing curve-skeletons with medial geodesic function. In *Symposium on geometry processing*, volume 6, pages 143–152.

Gobeawan, L., Lin, E., Tandon, A., Yee, A., Khoo, V., Teo, S., Yi, S., Lim, C., Wong, S., Wise, D., et al. (2018). Modeling trees for virtual singapore: From data acquisition to citygml models. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 42.

Gudmundsson, J., Hammar, M., and van Kreveld, M. (2002). Higher order delaunay triangulations. *Comput. Geom.*, 23(1):85–98.

Guo, J., Xu, S., Yan, D.-M., Cheng, Z., Jaeger, M., and Zhang, X. (2018). Realistic procedural plant modeling from multiple view images. *IEEE transactions on visualization and computer graphics*.

Hackenberg, J., Morhart, C., Sheppard, J., Spiecker, H., and Disney, M. (2014). Highly accurate tree models derived from terrestrial laser scan data: A method description. *Forests*, 5(5):1069–1105.

Holmgren, J. and Persson, Å. (2004). Identifying species of individual trees using airborne laser scanner. *Remote Sensing of Environment*, 90(4):415–423.

Honda, H. (1971). Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body. *Journal of theoretical biology*, 31(2):331–338.

Hyyppä, J., Hyyppä, H., Inkinen, M., Engdahl, M., Linko, S., and Zhu, Y.-H. (2000). Accuracy comparison of various remote sensing data sources in the retrieval of forest stand attributes. *Forest Ecology and Management*, 128(1-2):109–120.

Hyyppa, J., Kelle, O., Lehikoinen, M., and Inkinen, M. (2001). A segmentation-based method to retrieve stem volume estimates from 3-d tree height models produced by laser scanners. *IEEE Transactions on geoscience and remote sensing*, 39(5):969–975.

Kamal, M., Phinn, S., and Johansen, K. (2015). Object-based approach for multi-scale mangrove composition mapping using multi-resolution image datasets. *Remote Sensing*, 7(4):4753–4783.

Ke, Y. and Quackenbush, L. J. (2011). A review of methods for automatic individual tree-crown detection and delineation from passive remote sensing. *International Journal of Remote Sensing*, 32(17):4725–4747.

Lau, A., Martius, C., Bartholomeus, H., Shenkin, A., Jackson, T., Malhi, Y., Herold, M., and Bentley, L. P. (2019). Estimating architecture-based metabolic scaling exponents of tropical trees using terrestrial lidar and 3d modelling. *Forest Ecology and Management*, 439:132–145.

Liang, X., Kankare, V., Hyyppä, J., Wang, Y., Kukko, A., Haggrén, H., Yu, X., Kaartinen, H., Jaakkola, A., Guan, F., et al. (2016). Terrestrial laser scanning in forest inventories. *ISPRS Journal of Photogrammetry and Remote Sensing*, 115:63–77.

Livny, Y., Pirk, S., Cheng, Z., Yan, F., Deussen, O., Cohen-Or, D., and Chen, B. (2011). *Texture-lobes for tree modelling*, volume 30. ACM.

Livny, Y., Yan, F., Olson, M., Chen, B., Zhang, H., and El-Sana, J. (2010). Automatic reconstruction of tree skeletal structures from point clouds. In *ACM Transactions on Graphics (TOG)*, volume 29, page 151. ACM.

Maltamo, M., Næsset, E., and Vauhkonen, J. (2014). Forestry applications of airborne laser scanning. *Concepts and case studies. Manag For Ecosys*, 27:460.

Markku, Å., Raumonen, P., Kaasalainen, M., and Casella, E. (2015). Analysis of geometric primitives in quantitative structure models of tree stems. *Remote Sensing*, 7(4):4581–4603.

Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

Naesset, E. (1997). Estimating timber volume of forest stands using airborne laser scanner data. *Remote Sensing of Environment*, 61(2):246–253.

Nan, L. (2018). Mapple. https://3d.bk.tudelft.nl/liangliang/software.html.

Nan, L. (2019). Easy3d. https://github.com/LiangliangNan/Easy3D.

Ogniewicz, R. L. and Kübler, O. (1995). Hierarchic voronoi skeletons. *Pattern recognition*, 28(3):343–359.

Olofsson, K., Holmgren, J., and Olsson, H. (2014). Tree stem and height measurements using terrestrial laser scanning and the ransac algorithm. *Remote sensing*, 6(5):4323–4344.

Pinz, A. J. (1991). A computer vision system for the recognition of trees in aerial photographs.

Pollock, R. (1998). Individual tree recognition based on a synthetic tree crown image model. In *Proc. of the International Forum on Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry*, pages 25–34. Victoria, British Columbia, Canada.

Quan, L., Tan, P., Zeng, G., Yuan, L., Wang, J., and Kang, S. B. (2006). Image-based plant modeling. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 599–604. ACM.

Raumonen, P., Kaasalainen, M., Åkerblom, M., Kaasalainen, S., Kaartinen, H., Vastaranta, M., Holopainen, M., Disney, M., and Lewis, P. (2013). Fast automatic precision tree models from terrestrial laser scanner data. *Remote Sensing*, 5(2):491–520.

Reche-Martinez, A., Martin, I., and Drettakis, G. (2004). Volumetric reconstruction and interactive rendering of trees from photographs. In *ACM transactions on graphics (ToG)*, volume 23, pages 720–727. ACM.

Shlyakhter, I., Rozenoer, M., Dorsey, J., and Teller, S. (2001). Reconstructing 3d tree models from instrumented photographs. *IEEE Computer Graphics and Applications*, 21(3):53–61.

Singh, K., Sohlberg, S., Sokolov, V., et al. (1986). Conceptual framework for the selection of appropriate remote sensing techniques.

Verroust, A. and Lazarus, F. (1999). Extracting skeletal curves from 3d scattered data. In *Proceedings Shape Modeling International'99. International Conference on Shape Modeling and Applications*, pages 194–201. IEEE.

Wang, D., Hollaus, M., Puttonen, E., and Pfeifer, N. (2016). Automatic and self-adaptive stem reconstruction in landslide-affected forests. *Remote Sensing*, 8(12):974.

Wu, S.-T. and Marquez, M. R. G. (2003). A non-self-intersection douglas-peucker algorithm. In *16th Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2003)*, pages 60–66. IEEE.

Xie, K., Yan, F., Sharf, A., Deussen, O., Huang, H., and Chen, B. (2016). Tree modeling with real tree-parts examples. *IEEE transactions on visualization and computer graphics*, 22(12):2608–2618.

Xu, H., Gossett, N., and Chen, B. (2007). Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Transactions on Graphics (TOG)*, 26(4):19.

Yan, D.-M., Wintz, J., Mourrain, B., Wang, W., Boudon, F., and Godin, C. (2009). Efficient and robust reconstruction of botanical branching structure from laser scanned points. In *2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics*, pages 572–575. IEEE.

Zhen, Z., Quackenbush, L., and Zhang, L. (2016). Trends in automatic individual tree crown detection and delineation—evolution of lidar data. *Remote Sensing*, 8(4):333.

Zhong, C., Malinen, M., Miao, D., and Fränti, P. (2015). A fast minimum spanning tree algorithm based on k-means. *Information Sciences*, 295:1–17.

Zhou, H., Shenoy, N., and Nicholls, W. (2001). Efficient minimum spanning tree construction without delaunay triangulation. In *Proceedings of the 2001 Asia and South Pacific Design Automation Conference*, pages 192–197. ACM.

A POTENTIAL APPLICATIONS

Specific details of the potential applications of accurate tree models are provided in this appendix. Figure A.1 shows that important tree attributes such as the tree height and the trunk thickness can be measured from the reconstructed model. Figure A.2 indicates that the wood volume of the tree can also be estimated as the sum of the volume of all branch cylinders within the tree model. Figure A.3 gives an example of how the tree models help to enhance the realism of the digital urban scenes.

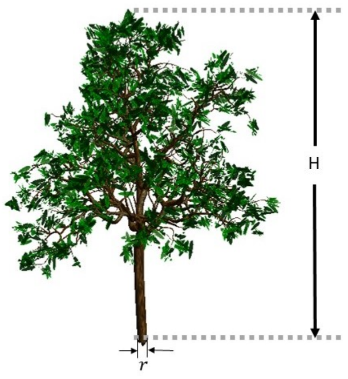


Figure A.1: Measurements of tree height and tree thickness.

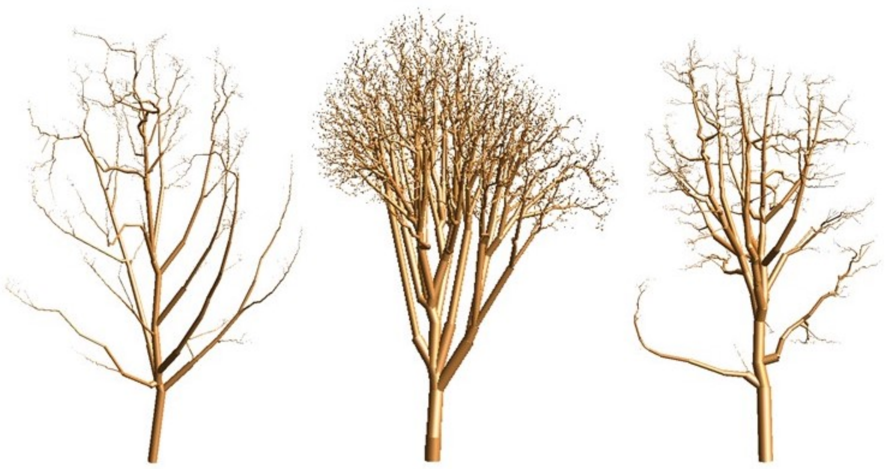


Figure A.2: Wood estimation.

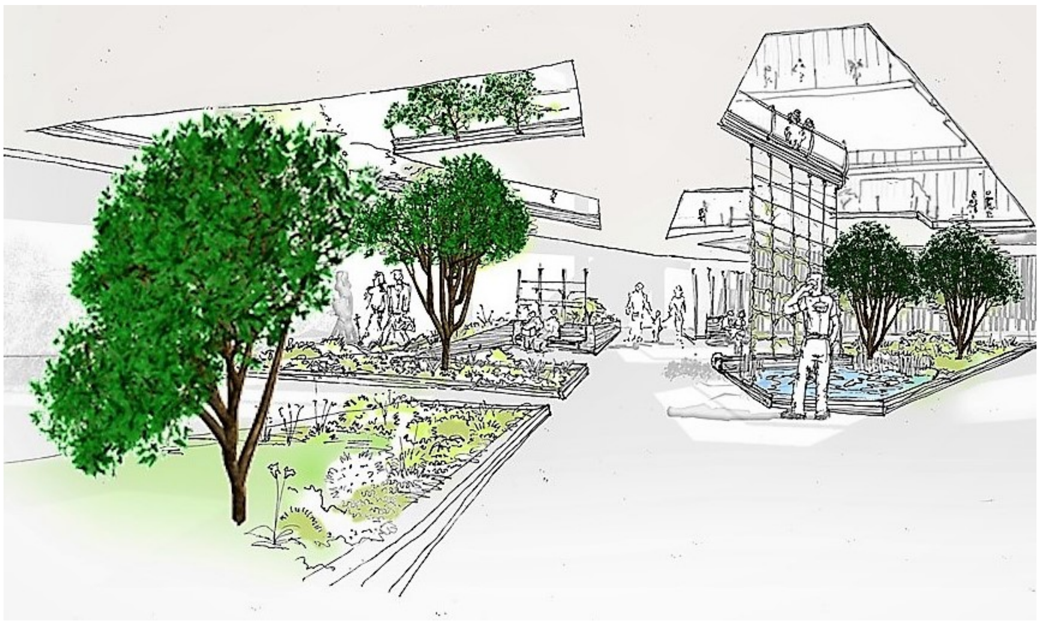


Figure A.3: An illustration how tree models enhance the realism of the scene.

COLOPHON

This document was typeset using LaTeX. The document layout was generated using the `arsclassica` package by Lorenzo Pantieri, which is an adaption of the original `classicthesis` package from André Miede.