# Revealing the skeleton from imperfect point clouds

**Alexander Klaus Bucksch**

# Revealing the skeleton from imperfect point clouds

**Proefschrift**

ter verkrijging van de graad doctor,

aan de Technische Universiteit Delft,

op gezag van de Rector Magnificius prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties

in het openbaar te verdedigen op donderdag 21 april 2011 om 10:00 uur

door

Alexander Klaus BUCKSCH

Master of Science in Informations- und Medientechnik,
Brandenburgische Technische Universität Cottbus, Duitsland

geboren te Böblingen, Duitsland.

Dit proefschrift is goedgekeurd door de promotor: Prof. Dr. M. Menenti

Copromotor: Dr. R. Lindenbergh

Samenstelling promotiecommissie:

| | | |
|---|---|---|
| Rector magnificius | - | voorzitter |
| Prof. Dr. M. Menenti | - | Technische Universiteit Delft, promotor |
| Dr. R. Lindenbergh | - | Technische Universiteit Delft, copromotor |
| Prof. Dr. N. Pfeifer | - | Technische Universität Wien |
| Prof. Dr. R. Veldtkamp | - | Universiteit Utrecht |
| Prof. Dr. P. van Oosterom | - | Technische Universiteit Delft |
| Dr. M. Spagnuolo | - | IMATI, Genua, Italië |
| Dr. S. Fleck | - | Georg-August Universität Göttingen |
| Prof. Dr. R. Hanssen | - | Technische Universiteit Delft, reservelid |

Copyright ©2011 by Alexander Bucksch All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic, mechanical, including photocopying, recording or by any information storage and retrieval system, without the prior permission of the author.

ISBN 978-3-86853-877-9

Typeset by the author with the LaTeXDocumentation System.
Author email: a.k.bucksch@tudelft.nl or a.bucksch@gmail.com

# Preface

Many people contributed in different ways to my thesis. As absolute Nr.1 there has to be named Roderik Lindenbergh who I met already on the first day when I stepped out of the train in Delft. He was a postdoc at this time and became something like my unofficial advisor. Later on he became my official advisor and finally co-promotor. And in all difficulties I faced he was the only person that was present throughout the whole PhD. I especially appreciated that he never restricted me in my research.

Massimo Menenti, who became my promoter when he was appointed as a professor at the TU Delft. I always liked the dualism that he structured a thesis about structure while learning to understand what I think about structure. I value it a lot, that he put a lot of effort into a topic, which was new to him. Also, I want to thank him for the very sharp comments about the content of the thesis.

Furthermore, I want to say thanks to Stefan Fleck for the opportunity to start up this thesis within the canopy analyzer project. I always enjoyed the visionary discussions we had especially during field work.

Ben Gorte who was supporting me a lot when he was the acting chair of OLRS. When things get more difficult than expected I often think: 'What would Ben do?'

Norbert Pfeifer who was my first supervisor. Although he left a few months after starting up the project he kept on supporting me. I appreciated and value a lot that whenever we met somewhere, he spent some time with me on scientific topics.

I further want to acknowledge the company Zoller und Fröhlich for financing a fellowship during the canopy analyzer project and Michael Suppa from the German Aerospace Center for financing a follow-up project to this thesis.

To update the thesis acknowledgment of Martin Kodde: I'm happy that we could extend the coffee tasting experiment within the faculty to a whole South-Holland wide pizza evaluation. These pizza dinners were always truly inspiring events for me. Also I want to acknowledge Gianni Orlando and Matthijs Stinissen for all the great times we had here and the lively friendship. Another important person in the last year of my PhD period became Daniel Berio whom I thank for making my thesis cover and many nights out. Also I want to send out a big thanks to all the guys from and around the CVO in Cottbus for their amazing ability to conserve long distance friendships.

Not being unnamed should be our 'Big Boss' Lidwien De Jong who greatly organizes the section and supports everybody with huge enthusiasm. A special thanks goes also to my office mates Carla Nardinocci, Hamid Ghrafarian and all other colleagues here at TU Delft.

Last but not least, thanks to my whole family, from parents to grand cousins, and all here unnamed friends for the support in the last years.

# Contents

**Summary** **ix**

**Samenvatting** **xiii**

**Zusammenfassung** **xvii**

**1 Introduction** **1**
- 1.1 General problem description . . . . . 1
- 1.2 Objectives . . . . . 7
- 1.3 Methodology in a nutshell . . . . . 8
- 1.4 Organization of the thesis . . . . . 9

**2 Skeleton extraction requirements** **11**
- 2.1 General problem description . . . . . 11
- 2.2 Descriptive and algorithmic skeleton requirements . . . . . 15
  - 2.2.1 Descriptive skeleton requirements . . . . . 15
  - 2.2.2 Algorithmic skeleton requirements . . . . . 16
- 2.3 Topology Preservation . . . . . 16
- 2.4 Metric representation . . . . . 18
- 2.5 Handling of imperfect data . . . . . 19
- 2.6 Computational efficiency . . . . . 20
- 2.7 Summary . . . . . 21

**3 Skeleton algorithm classes** **23**
- 3.1 Introduction . . . . . 23
- 3.2 2D Skeleton . . . . . 25
  - 3.2.1 Morphological thinning . . . . . 25
  - 3.2.2 Distance Transform . . . . . 27
  - 3.2.3 Voronoi Diagram methods . . . . . 29
- 3.3 1D Skeleton . . . . . 30
  - 3.3.1 Clustering Methods . . . . . 30
  - 3.3.2 Level set extraction . . . . . 31
  - 3.3.3 Graph reduction . . . . . 33
- 3.4 Summary . . . . . 35

**4 SkelTre Skeletonization 37**
4.1 Algorithm in a nutshell . . . . . . . . . . . . . . . . . . . . . . . . . 37
4.2 Octree-graph extraction . . . . . . . . . . . . . . . . . . . . . . . . . 38
4.2.1 Octree generation . . . . . . . . . . . . . . . . . . . . . . . . 38
4.2.2 Extraction and labeling of the octree-graph . . . . . . . . . . 39
4.3 Graph reduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 41
4.3.1 Topological Preservation and Metric Representation . . . . . 45
4.4 Graph embedding . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
4.5 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51

**5 Analysis of the SkelTre Algorithm 53**
5.1 Algorithm Efficency . . . . . . . . . . . . . . . . . . . . . . . . . . . 53
5.1.1 Computational Complexity . . . . . . . . . . . . . . . . . . . 55
5.1.2 Algorithm behavior . . . . . . . . . . . . . . . . . . . . . . . 57
5.2 Results and Practical Validation . . . . . . . . . . . . . . . . . . . . 58
5.2.1 Trees . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58
5.2.2 Non-tree objects . . . . . . . . . . . . . . . . . . . . . . . . . 61
5.2.3 Comparison with CAMPINO . . . . . . . . . . . . . . . . . . 62
5.2.4 Behavior under rotations . . . . . . . . . . . . . . . . . . . . 64
5.2.5 Running time . . . . . . . . . . . . . . . . . . . . . . . . . . 65
5.3 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 68

**6 Extraction of branch length and diameter of trees 69**
6.1 Extracting branch length and diameter from terrestrial laser scanning point clouds of orchard trees . . . . . . . . . . . . . . . . . . . . . . . 70
6.1.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 70
6.1.2 Data collection and preprocessing . . . . . . . . . . . . . . . 71
6.1.3 Methods for tree parameter extraction . . . . . . . . . . . . . 72
6.1.4 Results and Discussion . . . . . . . . . . . . . . . . . . . . . 77
6.1.5 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
6.2 Delineation and stem diameter estimation from Airborne Laser Altimetry 82
6.2.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 82
6.2.2 FLI-MAP 400 . . . . . . . . . . . . . . . . . . . . . . . . . . 82
6.2.3 Species dependent measurement of the diameter at breast height 83
6.2.4 Methodology . . . . . . . . . . . . . . . . . . . . . . . . . . . 84
6.2.5 Results and Validation . . . . . . . . . . . . . . . . . . . . . 88

6.2.6 Simulated data . . . . . . . . . . . . . . . . . . . . . . . . 88
6.2.7 Four test cases . . . . . . . . . . . . . . . . . . . . . . . . 92
6.2.8 Delineated forest trees . . . . . . . . . . . . . . . . . . . . . 93
6.2.9 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . 96
6.3 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 96

**7 Conclusions 99**
7.1 Summary of the results . . . . . . . . . . . . . . . . . . . . . . . 99
7.2 Outlook and future work . . . . . . . . . . . . . . . . . . . . . . 102
7.2.1 New methodology . . . . . . . . . . . . . . . . . . . . . . . 102
7.2.2 Tree and forest applications . . . . . . . . . . . . . . . . . . 103
7.2.3 Other applications . . . . . . . . . . . . . . . . . . . . . . . 104
7.2.4 A far away dream . . . . . . . . . . . . . . . . . . . . . . . 104
7.2.5 And what about the leaves? . . . . . . . . . . . . . . . . . . 104

**A Glossary and Symbols 107**

**References 110**

# Summary

## Revealing the skeleton from imperfect point clouds

Quantifying our surrounding environment in terms of sizes and orders has always been of interest, because it enables us to visualize, describe and interpret our environment. In the last decade terrestrial laser scanners became available as a tool to measure objects in our surrounding environment. Terrestrial laser scanning samples surfaces with millions of 3D points to be stored as a point cloud. These point clouds contain information of sizes and orders of the sampled objects.

Recently, trees became an object of high interest, because they contain a rich and complex structure, which is to be exploited in terms of branch length, branch diameter and branching hierarchy. Such information can be used for forestry, hydrology, ecology and visualization applications.

This thesis introduces the SkelTre algorithm (*Skel*etonization of *Tre*es) as a method to extract a structure description from point clouds. Such a structure description is a so-called skeleton which is similar to a collection of connected lines. A skeleton enables us to extract the branching hierarchy and the size parameters of tree branches from point clouds. Laser scanned point clouds contain only information about the surface of a scanned object and no information about its inside. One benefit of SkelTre is its ability to extract a skeleton purely from the surface samples represented in the point cloud.

The SkelTre algorithm meets four main requirements explicitly:

1. Topological preservance - to enable the branching hierarchy extraction
2. Metric representation - to enable the measurement of sizes
3. Imperfect data handling - to address data characteristics such as noise, underampling and varying point density
4. Computational efficiency - to address the need for fast data processing on the given huge amount of data

The SkelTre algorithm treats all four requirements based on one single characteristic, the local elongation direction. This elongation direction describes locally how an object surface is sizing in a 3D space. A local elongation direction can be extracted from the given imperfect data by analyzing small cubical cells subdividing the point cloud into a 3D raster. The cubical cells are derived from a so-called octree subdivision. For each cell it is analyzed which side of the cell is passed by the point cloud. The passed sides are representing the local elongations which are further represented by a graph. In this graph, each cell is corresponding to a vertex. If cells share a passed side, than the corresponding vertices are linked by an edge.

The input point cloud is imperfect, because it contains noise, undersampling and varying point densities. Therefore, SkelTre treats these point cloud characteristics explic-

itly, to produce a result that is correctly representing the branching hierarchy. Therefore, before the graph is reduced to the skeleton a robustness criterion is applied. This criterion removes and adds edges to the graph. Edges are added on locations where data is missing because of undersampling or removed when noise causes unintended edges. At this stage the graph ideally represents the surface of an object and is to be reduced to a skeleton.

This corrected graph is then reduced to a skeleton by a set of newly developed rules. The reduction uses two suited vertex configurations occurring in the graph, so-called E-Pairs and V-Pairs. Basically the reduction relies on the property, that whenever an E-Pair is found and reduced, one or more V-Pairs are generated. Processing of a V-Pair leads to either new V-Pairs or otherwise a new E-Pair has to be found in the graph. If neither an E-Pair or a V-Pair is present in the graph it is ensured that the result is a skeleton. The resulting skeleton is related to a known topological structure, the Reeb-Graph.

During the reduction of the graph, the graph is embedded into the point cloud such that every vertex is locally centered within the point cloud. The correct branching order and centeredness enables the navigation to a chosen location in the point cloud to derive a branch diameter. A first implementation of the algorithm is given to demonstrate its efficiency on huge point cloud data. On many examples it is evaluated, that all four requirements are met. These examples include non-tree objects to emphasize the generality of the introduced skeletonization framework.

The practical use of the SkelTre algorithm is demonstrated on two applications demanding automatic size measurements on trees. In contrast to older methods, both applications do not rely on pre-knowledge of the tree species. Hence, the assessment of trees in these applications is purely based on the point cloud data. Both applications use the newly developed HARPER method to estimate the diameter under support of the SkelTre skeleton. The HARPER method estimates the diameter of a branch on the basis of the double distance of the point cloud points to the skeleton. The length of branches is derived as the sum of the edge lengths in the skeleton of a branch.

The first application is the automatic estimation of the tree parameters branch length and branch diameter from laser scanned orchard trees. The estimated parameters are a key to understand a number of physiological processes in the tree canopy. Furthermore, these parameters are useful to determine the vitality of a tree. Orchard trees are optimized by branch cuttings to maximize the crop load on the tree. These cuttings destroy the growth pattern of the tree. Therefore, assumptions about the growth pattern can not be made for the estimation of the branch sizes. For these orchard trees the branch length and diameter was derived with the HARPER method. High correlations above 0.9 between field and automatic measurement were found in the frequency distributions of the branch length and diameter classes. These results show that automatic measurement of tree parameters from terrestrial laser scans is possible.

The second application demonstrates a possible future use of the SkelTre skeleton. Here, the skeletonization was applied to airborne obtained laser point clouds to estimate the trunk diameter at breast height (1,30m). The diameter at breast height is one key parameter for calculating hydrological roughness in flooding areas and to estimate the woody volume of a tree. The density of airborne obtained point clouds is only 75 points per square meter. Still, the trunk diameter at breast height could be extracted. On simulated airborne scanned trees a correlation of 0.97 was achieved between the diameter at breast height of the known simulated trees and the HARPER method. Furthermore, test cases were extracted from the airborne obtained point cloud of the recently updated Actual Height Model of the Netherlands, AHN2. On these test cases the estimated diameter with the HARPER method deviated less than a standard cylinder fitting method from manually measured diameters in the field. Still, the validation of a whole forest remains difficult, because not all trees of the forest are found back in the AHN2 point cloud. Such AHN2 point clouds will be available for the whole Netherlands in 2013 and other countries collect similar point clouds at this time. The wide availability of such point clouds in future underlines the relevance of this application.

# Samenvatting

## Skeletextractie uit imperfecte puntenwolken

Het kwantificeren van onze omgeving in termen van grootte en ordening is altijd al van groot belang geweest, want het stelt ons in staat om onze omgeving te visualiseren, te beschrijven en te interpreteren. In het afgelopen decennium kwamen terrestrische laserscanners beschikbaar als hulpmiddel om objecten in onze omgeving in te meten. Terrestrische laserscanners leggen oppervlaktes vast met miljoenen 3D punten die opgeslagen worden als een puntenwolk. Deze puntenwolken bevatten informatie over afmetingen en ordening van de ingewonnen objecten.

Met name bomen zijn recentelijk onderwerp van studie geworden, omdat ze een veelzijdige en complexe structuur hebben. Met behulp van deze structuur kunnen grootheden zoals de lengte en de diameter van takken bepaald worden, maar ook de vertakkingshiërarchie. Dergelijke informatie kan gebruikt worden in de bosbouw, de hydrologie, de ecologie en voor visualisatie.

In dit proefschrift wordt het SkelTre algoritme geïntroduceerd als een methode voor het extraheren van een structuurbeschrijving uit puntenwolken. Deze structuurbeschrijving wordt een skelet genoemd en op een verzameling van lijnen waarmee een object wordt beschreven. Een skelet stelt ons in staat om automatisch de aftakkingshiërarchie en de afmetingen van de takken en de stam te bepalen uit met name laserscandata. SkelTre is de afkorting van *Skeletonization of Trees*. Omdat de puntenwolken ingewonnen worden uit afstandsmetingen tot het oppervlak van een object, en niet tot in een object, is het skeletonisatie-algoritme zodanig ontwikkeld dat het werkt op basis van deze bemonsterde oppervlaktes.

Het SkelTre algoritme is zo ontworpen dat aan vier belangrijke randvoorwaarden wordt voldaan:

1. Behoud van topologie - om het bepalen van aftakkingshiërarchie mogelijk te maken
2. Metrische representatie - om het bepalen van afmetingen van takken en stam mogelijk te maken
3. Omgang met fouten in de data - om ondanks imperfecties toch een zo goed mogelijk resultaat te verkrijgen
4. Rekenkundige efficiëntie - om grote hoeveelheden data snel kunnen te verwerken

In het SkelTre-algoritme wordt met alle vier randvoorwaarden rekening gehouden door een enkele karakteristiek: de lokale lengterichting. Deze lokale lengterichting beschrijft lokaal de dimensionering van een object in de 3D-ruimte. Een lokale lengterichting kan worden bepaald uit een puntenwolk, inclusief gebreken, door te analyseren hoe de puntenwolk omsloten wordt door kleine kubusvormige cellen. Alle cellen samen vormen een 3D raster en worden gerepresenteerd door een zogenaamde 3D-octree onderverdeling. Van elke cel, wordt bepaald welke zijden worden doorsneden door de puntenwolk. De doorsneden zijdes representeren de lokale lengterichtingen, die vervolgens worden vastgelegd in een graaf. De knopen van deze graaf corresponderen met de cellen. Twee knopen worden door een kant verbonden als de puntenwolk de corresponderende zijdes van de cellen doorsnijdt.

De invoergegevens, dat wil zeggen de puntenwolk, bevatten de nodige gebreken: ze zijn behept met ruis, zijn niet altijd helemaal dekkend en als ze dekken, varieert vaak nog de puntdichtheid. In SkelTre wordt speciaal met deze gebreken rekening gehouden, zodat het eindresultaat de aftakkingshiërarchie op een juiste wijze weergeeft. De ruis in de data wordt met name aangepakt door een zogenaamd robuustheidscriterium, waarbij kanten uit de graaf verwijderd dan wel toegevoegd worden op die locaties waar de ruis in de data onbedoelde delen van het object vormt, of waar juist data mist. Op dit punt in het proces representeert de graaf de oppervlakte van het object, waarna het gereduceerd tot een skelet.

De graaf wordt door middel van een aantal regels gereduceerd tot het gewenste skelet. Deze regels verwijderen de overbodige kanten door systematisch op zoek te gaan naar twee geschikte knopen-configuraties die in de graaf voorkomen, de zogenoemde E-Paren en V-Paren. In principe is deze reductie gebaseerd op de eigenschap dat zodra een E-Paar gevonden en verwerkt is volgens de geldende regels, een of meerdere V-Paren aangemaakt worden. De verwerking van een V-Paar leidt altijd tot ofwel een nieuw V-Paar, dan wel een herhaalde zoekactie naar het volgende E-Paar. Dit principe garandeert dat de gewenste reductie ook daadwerkelijk plaats vindt.

De reductieprocedure verzekert dat het resultaat een skelet is en is gerelateerd aan een bekende topologische structuur, de Reeb-Graaf. Gedurende de reductie van de graaf, wordt de link tussen de veranderende graaf en puntenwolk bijgehouden, waardoor op het eind elk punt van de graaf op de juiste locatie binnen de puntenwolk geplaatst kan worden. Daardoor kan het skelet gebruikt worden om een geschikt stuk scan data te selecteren voor bijvoorbeeld het bepalen van diameters. Een en ander wordt geillustreerd aan de hand van een eerste implementatie van het SkelTre-algoritme. Dankzij deze implementatie kan de efficiëntie van het algoritme gedemonstreerd worden en is het mogelijk de correctheid in de praktijk te valideren aan de hand van enkele voorbeeld-puntenwolken. Het SkelTre algoritme wordt ook gedemonstreerd op puntenwolken waarin objecten anders dan bomen vastgelegd zijn. Hiermee wordt benadrukt dat de procedures en algoritmes die hier gepresenteerd worden meer algemeen toepasbaar zijn.

De praktische toegevoegde waarde wordt gedemonstreerd aan de hand van twee toepassingen om waarbij boomparameters worden bepaald. In tegenstelling tot andere, oudere methoden, is het hierbij niet noodzakelijk dat van te voren bekend is wat voor soort boom geanalyseerd wordt. Beide toepassingen maken gebruik van de nieuw ontwikkelde HARPER-methode die de diameter van een boom bepaalt op basis van het SkelTre skelet. HARPER schat de diameter van een tak uit de dubbele afstand van een geschikt stuk van de puntenwolk tot het skelet, terwijl de lengte van de takken bij eerste benadering gegeven wordt door een som van geschikte kanten uit het skelet.

In de eerste toepassing werd het algoritme toegepast op bomen uit een boomgaard. Bomen in een boomgaard zijn door mensen geoptimaliseerd om de hoeveelheid vruchten in de boom te maximaliseren. Deze optimalisatie heeft echter tot effect dat het natuurlijk groeipatroon van deze bomen niet meer zichtbaar is, als gevolg van het steeds verwijderen van ongewenste takken. Vanwege de missende takken is het dus ook niet goed mogelijk om aannames te doen over de aftakkingshiërarchie. Van deze bomen werden de lengtes en diameters van takken geschat op grond van hun SkelTre skeletten. Het bleek dat de frequentiedistributie van traditionele veldmetingen en de nieuwe automatische SkelTre metingen sterk op elkaar leken. Dit toont aan dat met deze nieuwe methode lengtes en diameters van takken efficient en nauwkeurig bepaald kunnen worden zonder afhankelijk te zijn van handmetingen. Zulke automatische schattingen zijn de sleutel tot een beter begrip van enkele fysiologische processen in boomkruinen die de gezondheid van bomen uitdrukken.

De tweede toepassing demonstreert een mogelijk toekomstig gebruik van het SkelTre skelet. Hierbij werd de skeletonisatie toegepast op vanuit de lucht ingewonnen laser data van bomen om de diameter op borsthogte te schatten. De diameter op borsthoogte (1,30 m hoogte boven het maaiveld) is een zeer belangrijke parameter voor het berekenen van hydrologische ruigheid in overstromingsgebieden en voor het bepalen van het houtvolume in een boom. De dichtheid van vanuit de lucht ingewonnen laser data is met 75 punten per vierkantemeter. Alhoewel deze puntdichtheid veel lager is dan die van terrestrische puntenwoloken, was het toch mogelijk de dikte van de stam van een boom te bepalen op borsthoogte. In een gesimuleerde dataset werd een correlatie van 0.97 behaald tussen gesimuleerde boomdiktes, en boomdiktes bepaald uit de SkelTre skeletten. Bij echte data leek de HARPER methode ook goed te werken, maar bleek het wel moeilijk om de resultaten te valideren, doordat de handgemeten bomen soms lastig waren terug te vinden in de laser data set. Voor dit praktijkvoorbeeld werden puntenwolken uit het recent bijgewerkte Actueel Hoogtemodel van Nederland, AHN2, toegepast. Deze data is na 2013 voor geheel Nederland beschikbaar, terwijl andere landen ook al bezig zijn met het inwinnen van nationale laser data sets. Dit laat zien dat automatische methodes die boomparameters kunnen schatten uit laser scan data breed kunnen worden toegepast in de komende jaren.

# Zusammenfassung

## Skelettextraktion aus imperfekten Punktwolken

Von jeher ist die Beschreibung von Größen und Strukturen in unserer Umgebung von großer Bedeutung, da es uns in die Lage versetzt unsere Umgebung darzustellen, zu beschreiben und zu interpretieren. Im letzten Jahrzehnt wurden terrestrische Laserscanner zum Messen unserer Umgebung verfügbar. Das Besondere an ihnen ist, daß sie Millionen von Punkten auf Oberflächen in ihrer Umgebung messen und speichern. Diese Punkte bilden eine 3D Punktwolke, die Information über die Größe und Struktur eines eingemessenen Objektes enthält.

Bäume haben zunehmend das Interesse als automatisch zu vermessendes Studienobjekt geweckt, weil sie eine komplexe Kronentruktur besitzen. Unter zur Hilfenahme der Struktur können Größen wie Länge und Durchmesser von  Ästen und Stämmen bestimmt werden. Gleichermaßen kann auch die Verzweigungshierarchie erfasst werden. Diese Größen finden ihre Anwendung in der Forstwirtschaft, der Hydrologie, der Ökologie und in der Visualisierung.

In dieser Dissertation wird ein neuartiger Algorithmus (SkelTre-Algorithmus) zur Extraktion einer Strukturbeschreibung aus Punktwolken vorgestellt. Diese Strukturbeschreibung wird Skelett genannt und beschreibt ein Objekt linienartig. Die Abkürzung SkelTre stammt aus dem Englischen und bedeutet soviel wie Skelettierung von Bäumen (Skeletonization of Trees). Ein mit SkelTre extrahiertes Skelett erlaubt in Kombination mit der ursprünglichen Punktwolke die Bestimmung der Astgrößen und der Asthierarchie. Punktwolken die mit terrestrischen Laserscannern erfaßt wurden, bestehen aus Messungen von Objektoberflächen und repräsentieren somit nicht das Innere eines Objektes. Eine Besonderheit des SkelTre-Algorithmus ist, daß er das Skelett auf der Basis von Oberflächenmessungen bestimmen kann.

SkelTre berücksichtigt dabei vier Randbedingungen:

1. Topologieerhaltung - um die Bestimmung der Verastungshierarchie zu ermöglichen
2. Metrische Repräsentation - um das Bestimmen von Größen zu erlauben
3. Umgang mit fehlerhaften Daten - um trotz verschiedenster Datenfehler ein bestmögliches Resultat zu erzielen
4. Berechnungsgeschwindigkeit - um den Umgang mit großen Datenmengen zu erlauben.

Um alle vier Randbedingunegen zu erfüllen benötigt SkelTre nur eine Charakteristik: die lokale Elongationsrichtung. Die Elongationsrichtung beschreibt lokal die Ausdehnung des Objektes im dreidimensionalen Raum und kann auch in fehlerhaften Daten bestimmt werden. Die Bestimmung der Elongationsrichtung geschieht auf Grundlage kleiner kubischer Zellen, die in einem dreidimensionalen Raster angeordnet sind. Jede dieser Zellen wird daraufhin untersucht, ob die Punktwolke die Seiten der Zelle schneidet. Die geschnittenen Zellseiten repäsentieren die lokale Elongationsrichtung und werden als ein Graph wiedergegeben. Jede Zelle wird durch einen Vertex in diesem Graph repräsentiert und die Kanten des Graphen, welche die Vertices verbinden, repräsentieren die geschnittenen Zellseiten. Der resultierende Graph besteht somit aus Vertices, welche verbunden werden, wenn eine Zelle eine geschnittene Seite mit einer anderen Zelle teilt.

Die von einem Laserscanner erfaßte Punktwolke ist jedoch fehlerhaft: Sie ist verrauscht, enthält Löcher in gemessenen Oberflächen aufgrund von Überdeckungseffekten und besitzt eine unterschiedliche Punktdichte an verschieden Stellen des gemessenen Objektes. Kanten, die auf Grund von Rauschen erzeugt worden sind oder Kanten, die auf Grund von Überdeckungen fehlen werden durch ein Robustheitskriterium korrigiert. Dieses Kriterium fügt Kanten zu oder entfernt sie bevor die Reduktion zum Skelett stattfindet. An diesem Punkt repräsentiert der Graph idealerweise die Oberflache eines Objektes.

Der Graph wird anschließend durch neu entwickelte Regeln zu einem Skelett reduziert. Die Reduktion wird erlangt, indem zwei spezielle Vertex-Konfigurationen im Graphen gesucht werden. Diese beiden Konfigurationen heißen E-Pair und V-Pair. Im Prinzip wird erst ein E-Pair gesucht, welches die Eigenschaft besitzt bei Reduktion immer in einem V-Pair zu resultieren. Von einem V-Pair ist bekannt, daß seine Reduktion entweder in einem weiteren V-Pair resultiert oder anderenfalls ein neues E-Pair gefunden werden muss. Wenn weder ein E-Pair noch ein V-Pair im Graphen vorhanden ist, so ist garantiert, daß es sich um ein Skelett handelt. Dieses Skelett beinhaltet eine bekannte topologische Struktur, den sogenannten Reeb-Graphen.

Während der eigentlichen Reduktion werden die Vertices zentriert in Teile der Punktwolke eingebettet, um Durchmesser bestimmen zu können. Trotz der beschriebenen Störeinflüsse liegt das berechnete Skelett zentriert in der Punktwolke und gibt die Verastungshierarchie korrekt wieder. Das extrahierte Skelett ermöglicht es durch die Punktwolke zu navigieren und an frei wählbaren Stellen den Durchmesser von Ästen zu bestimmen.

Anhand von zahlreichen Beispielen wird die Qualität des Algorithmus evaluiert. Die Evaluierung legt ihren Schwerpunkt auf die Einhaltung der geforderten vier Randbedingungen. Das breite Einsatzspektrum des Algorithmus wird zusätzlich anhand von Beispielobjekten demonstriert, die keine Baumstruktur aufweisen. Eine erste Implemetierung des Algorithmus ist als Pseudocode gegeben um seinen effizienten Umgang mit grossen Punktwolkendaten darzustellen.

Anhand von zwei Anwendungsbeispielen konnte die Praxistauglichkeit des Algorithmus zur automatischen Bestimmung von Baumparametern im Kronen- und Stammraum aufgezeigt werden. Im Gegensatz zu älteren Methoden kann hierbei auf Vorkenntnisse der Wuchsform einer Baumsorte verzichtet werden. Beide Anwendungsbeispiele benutzen das SkelTre-Skelett und die neuentwickelte HARPER-Methode zur Bestimmung von Durchmessern. Die Durchmesserbestimmung basiert auf dem doppelten Abstand der Punkte zum Graphen. Die Astlänge ist im Skelett durch die Kantenlängen des Graphen wiedergegeben.

Die erste Anwendung beschreibt die automatische Messung von Obstbäumen, deren

Äste beschnitten werden, um die Ernteerträge zu maximieren. Auf Grund dieser Astbeschneidungen ist das natürliche Verastungsmuster des Baumes in der Punktwolke nicht mehr vollständig wiedergegeben. Deshalb ist es nicht möglich Annahmen über die Verastung in die Messung der Äste einzubringen. Für diese Bäume werden Längen und Durchmesser pro Ast automatisch bestimmt. Als Resultat wurde gezeigt, daß die automatische Messung auf Basis des Skelletes sehr starke Ähnlichkeit mit den Längen- und Durchmesserklassen der Feldmessung aufweist. Die Längen- und Durchmesserklassen erreichten Korrelationskoeffizienten von über 0.9. Dies bestätigt, daß automatisierte Messungen von Astgrössen möglich ist. Diese Messungen können zum besseren Verständnis der physiologischen Prozesse in Baumkronen verwendet werden und um die Vitaliät von Einzelbäumen zu erfassen.

Die zweite Anwendung untersucht einen zuküftigen Anwendungsbereich von SkelTre. In diesem Beispiel werden Baumstammdurchmesser in Punktwolken gemessen, die aus der Luft gewonnen wurden. Diese Punktwolken weisen eine deutlich geringere Punktdichte auf als terrestrisch erzeugte Punktwolken. Die hier verwendete Punktdichte beträgt 75 Punkte pro Quadratmeter. Es ist jedoch zu erwarten, daß diese Punktdichte in den kommenden Jahren weiter ansteigt. Trotz der viel geringeren Punktdichte ist es möglich den Stammdurchmesser auf 1.30 m Höhe zu bestimmen. Dieser sogenannte Brusthöhendurchmesser ist ein häufig verwendeter Parameter zur Berechnung der hydrologischen Rauhigkeit in Hochwassergebieten. Zuerst wurde der Durchmesser für Bäume aus einer bekannten Luftdatensimulation bestimmt. Hierbei wurde ein sehr guter Korrelationskoeffizient von 0.97 zwischen den bekannten Durchmessern der Simulationsdaten und den automatisch extrahierten Stammdurchmessern erreicht. Die Berechnung des Durchmessers auf echten Punktwolken zeigte auf, daß die HARPER-Methode in allen Fällen bessere Ergebnisse erzielte als eine halbautomatische Standardmethode. Die Validierung eines ganzen Waldes bleibt nach wie vor schwierig, da es nicht immer möglich ist jeden Baum im Wald in der Punktwolke wiederzufinden. Die benutzten Punktwolken sind Bestandteil des sogenannten Aktuellen Höhenmodells der Niederlande. Diese Punktwolken werden bis 2013 für die gesamten Niederlande verfügbar sein. Die weite Verfügbarkeit solcher Punktwolken in den Niederlanden und in absehbarer Zeit auch in anderen Ländern unterstreicht die praktische Relevanz dieser Anwendung.

# 1

# Introduction

## 1.1 General problem description

At the end of the seventies [Adams, 1979] described two major mistakes of humanity. Firstly leaving the trees as a habitant and secondly start wearing digital watches. Nowadays a third mistake can be observed in the spectator enthusiastic reaction while watching his first laser scan (Figure 1.2 and Figure 1.3). An example reaction can be:" Wow ... that's like a black and white digital photograph and even in 3D". Laser scans are still new to most non-experts and therefore not directly put into the right context. In fact the comparison of a laser scan with a digital photograph is intuitive and easy to understand. Basically a laser scan is a digital photograph taken from the panoramic surrounding of the laser scanner that contains intensity values. These intensity values are often represented in a greyscale. At the first place a laser scanner assigns a distance to every pixel in the picture, which enables positioning of each pixel in a 3D coordinate system. This distance is measured from the scanner itself to the first visible surface and determined by measuring the travel time of the laser from the object to the surface and back to the scanner. The instrument rotates around its vertical and horizon-

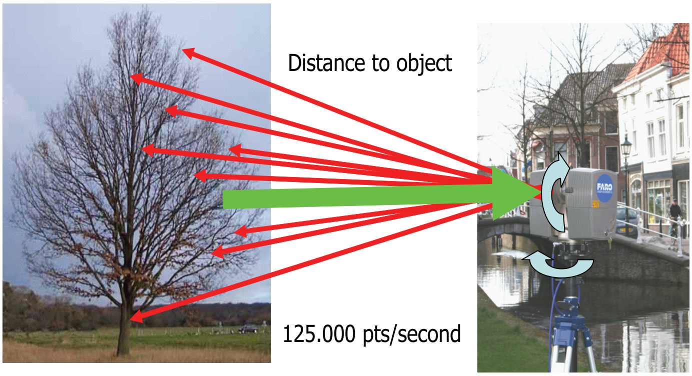


Figure 1.1: Simplified scanning principle of a terrestrial laser scanner

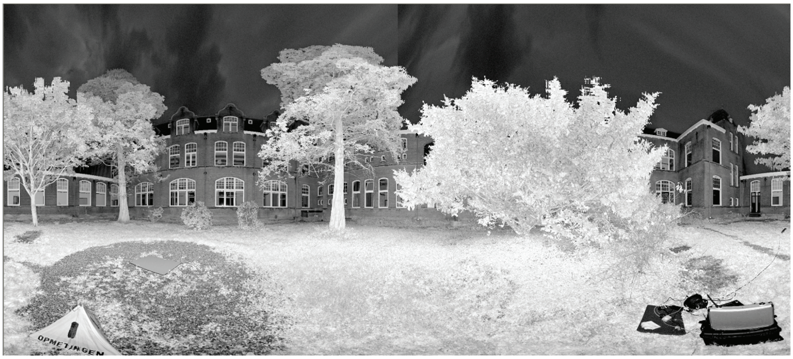


Figure 1.2: Intensity image of a panoramic laser scan

tal axis, while sending out the laser as illustrated in Figure 1.1. Such a scanner captures about 125.000 points per second on surfaces in its panoramic surrounding. The result of this measuring procedure is points located in a 3D space with associated intensity values. Such a collection of points, a so-called *point cloud*, is shown in Figure 1.3. A single scan still contains knowledge about the neighborhood of each point because of the scanners vertical and horizontal scanning directions. This order is lost as soon as several scans are aligned to each other. This alignment is called registration. Figure 1.3 is a point cloud registered from several single scans. The considered point clouds in this thesis are unorganized and their neighborhood is not known.

The first surprise to the spectator of a laser scan happens when you tell him that the black and white panoramic picture is significantly different from the one he knows from black and white photographs. The observed picture is the reflection in a very small part of the light spectrum and not the reflection as we recognize it in a black and white photograph derived from the whole visible spectrum of the light. Because of that, the picture of the reflection intensity of a laser scanner is representing some scaling of the intensity values measured by the laser scanner which is often represented in a greyscale picture. For example in Figure 1.4 a tree looks very whitish because of an applied scaling to the intensity values and is not comparable to a black and white photograph derived from the full spectrum of the visible light.

A critical spectator, may react slightly shocked if the laser scan is turned around in the 3D space as shown in Figure 1.5 (c). The novice viewer starts discovering that the point cloud is noisy and varies a lot in its point density over the scan, depending on the reflective properties and the surface granularity of the scanned objects. Furthermore, occlusions due to overlapping object parts can be observed. These occlusions are visible as gaps in the point cloud Figure 1.5 (b). The example shown in Figure 1.5 (a) is a scanned apple tree, which illustrates the problems of scanning complex structures. Complex structures like the tree crown contain many difficulties if one aims at the extraction of the branching structure. These difficulties are highlighted in Figure 1.6, where some branches are even visually not distinguishable.

Until now, it was assumed that the scan is taken with a photo camera-like instrument

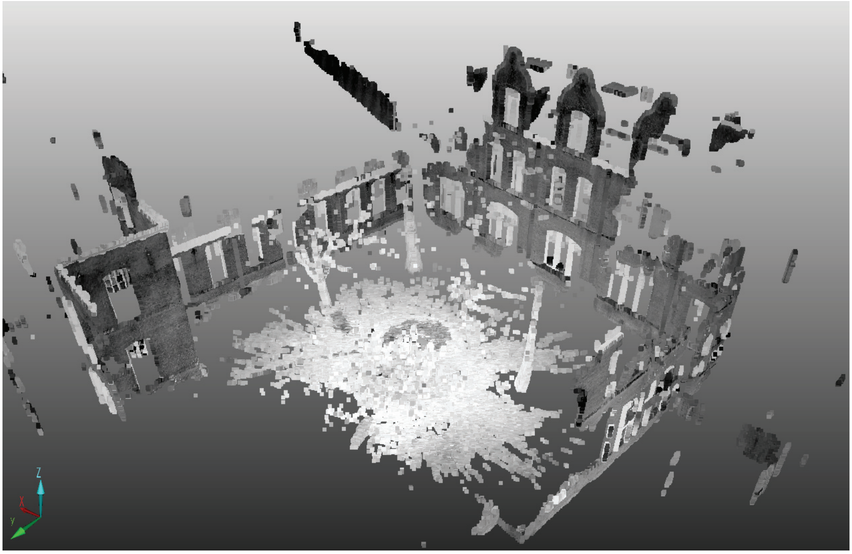


Figure 1.3: 3D view of a filtered laser scan. The scan is visualized with the standard software FARO Scene 5.7

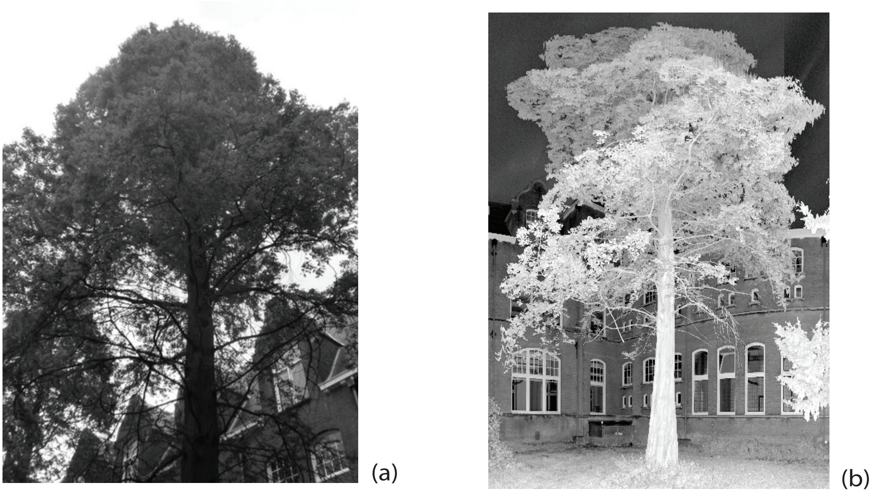


Figure 1.4: (a) Black and white photo of a tree and (b) the laser scan picture of the same tree

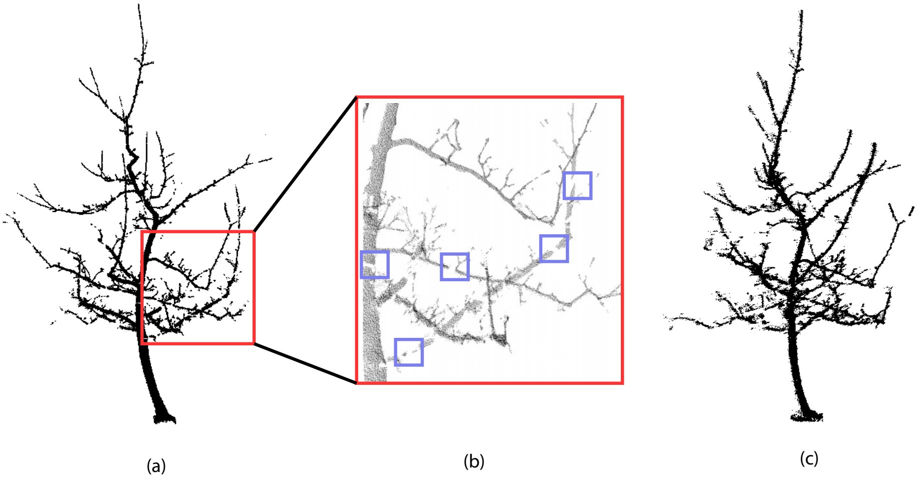


Figure 1.5: (a) Point cloud of a single terrestrial laser scan of a tree. (b) a magnification of the indicated tree part shown in (a). The marked regions show extreme cases of undersampling, visible as gaps in the point cloud. (c) The same point cloud as in (a), but rotated by 90 degree. It is blurred because of noise.

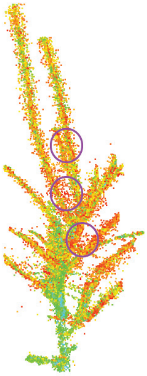


Figure 1.6: Enlarged point cloud part from a dense crown containing noise. The marked regions show noise masking the branching structure. Even for the human eye some branches are not distinguishable.

from the ground. Similar to a photo camera a laser scanner can also be mounted on an airplane or helicopter using almost the same scanning principle as shown before. A significant difference between airborne and terrestrial laser scanning is found in the point density. Suppose you take a photo from the air, you may not be able to distinguish between two objects, because the resolution of your camera may map the two objects on one pixel. This mapping behavior finds its reasoning in the larger area visible in the photograph, assuming that the resolution and zoom stay unchanged compared to the photo taken from the ground. Technically this is called undersampling and occurs in laser scans, too. The point density on the object surface decreases dramatically, and airborne obtained point clouds are subject to undersampling. These facts require to distinguish between terrestrial laser scanning from the ground and airborne laser scanning, as can be recognized in Figure 1.7. The tree in Figure 1.7(a) is the terrestrial laser scan with 526.745 points, and the tree in Figure 1.7(b) is the same tree scanned with an airborne system containing 15.460 points. The difference in point density is visible on the stem. Furthermore, the terrestrial scan was taken from the ground and shows individual branches, while the airborne laser was penetrating the tree from above the tree crown. Notably, we cannot recognize many details of the branching structure of the tree. In this thesis we are considering point clouds of real natural objects obtained by instruments measuring the dimensions of the object. Although problems are visualized on trees here, these problems are relevant in a general sense.

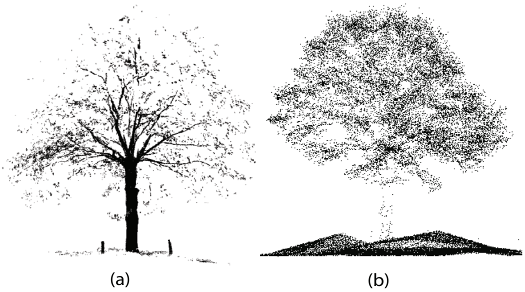


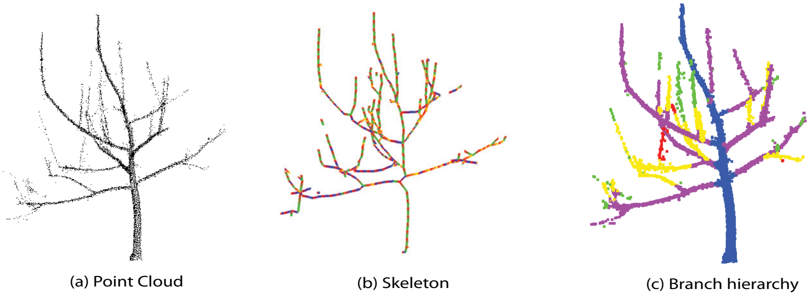
Figure 1.7: Comparison between the same tree scanned with (a) a terrestrial laser scanner, showing individual branches and high point density on the stem and (b) an airborne laser scanner showing almost none of the individual branches and less point density. The difference in point density is especially visible on the main stem of the tree.

One may characterize terrestrial laser scanning as scanning small areas with a locally high density of measurements (around $10^3$ points per square meter) and airborne laser scanning as measuring large areas with a low local density (up to 75 points per square meter). Still both methods produce huge data sets. The huge data sets are the reason why efficiency is an important factor for point cloud processing methods.

By looking at botanical trees, it is observable that their ramification is complicated. Their ramification has drawn a lot of attention. It was discovered that a relation between the ramification of a tree and its physiological processes, e.g. branch transpiration, exists, [Fleck, 2002]. The question we focus on is, whether laser scanning is a suitable technology to reveal the ramification of a tree, even though it is masked by noise and artifacts of imperfect point clouds. Therefore the aim is to develop a method that automatically extracts the structure of a tree, like its ramification, from a point cloud. On the basis of this structure description and its relation to the original point cloud, it is feasible to estimate tree parameters like branch length and diameter. In Chapter 6 of the thesis we will discuss applications where tree parameters are derived from an extracted ramification description. One of the easiest application examples is that the vitality of a tree is correlated to these extracted parameters [Fleck et al., 2004]. It may not demand the cutting of a tree to count the growth ring chronologies of the stem to determine its growth history [Roloff, 1986]. The missing gap between the interpretation of tree structure and the measurement of tree structure is its automatic extraction. Because of that, this thesis aims at the automatic extraction of a structure description.

Structure will be described in this thesis as follows: If we think again of a tree it is imaginable that it can be described by a line-like description representing the ramification of the tree. Such a description is called a *skeleton* and shown in Figure 1.8. Such a skeleton also contains the segmentation of the point cloud, e.g. into its branching hierarchy (Figure 1.8(c)). It should fulfill certain requirements to enable the applications described before. In informal words we may describe these requirements as follows:

*A skeleton preserves the order of defined object parts in a point cloud, it is centered within the point cloud and it is connected, whenever the object is connected.*


(a) Point Cloud

(b) Skeleton

(c) Branch hierarchy

Figure 1.8: (a) the point cloud, (b) an extracted skeleton from the point cloud, (c) a segmentation of the point cloud in (a) on the basis of the skeleton in (b). The same color corresponds to the same branching hierarchy.

The relation between physiological processes of trees on one hand, e.g. branch transpiration, and the manually measured structure on the other hand is already known, [Fleck, 2002]. So far, these relations have been validated only on manually extracted tree structures. As stated before, one long term goal is to relate such processes in a tree to an automatically extracted tree structure. This long term goal gives an application driven motivation to take a deeper look into the structure description we are going to extract from laser scanned trees and the applicability of skeletons to measure the parameters branch length and diameters based on the branching hierarchy. All these parameters are directly related to the wooden ramification of the tree, and therefore demand the measurement of the wooden ramification without leaves. During the last years I read many papers, suggesting that a skeleton may be suitable to extract tree parameters, but none of them stated what the extracted skeleton is actually representing. Some examples are [Gorte and Pfeifer, 2004], [Gorte, 2006], [Xu et al., 2007] and [Côtè et al., 2009]. All these references have in common that they use a skeleton to represent a tree, but none of them evaluates the used skeleton on the validity of its representation. Questions easily arise from this motivation:

1. Is the order of the branches extracted correctly?
2. How is the skeleton locally centered in the point cloud?
3. How does the method handle noise and other artifacts such as undersampling and varying point density?
4. Is the skeleton extraction fast enough to operate on huge point clouds obtained by laser scanners

As a result of this thesis an algorithm was developed that satisfies the needs for efficient computation on huge data sets and robustness against noise for application on laser scanning point clouds. The problem identified to develop such an algorithm is to find a framework which reduces the computational operations on the given huge amount of input data while preserving the validity of the skeleton in terms of branching order and centeredness as prerequisite. We assume that the concepts used in this algorithm are generally valid and because of that we also consider structures different then trees into the thesis in the evaluation of the algorithm.

## 1.2 Objectives

The main research question reads:

*Is it possible to automatically reveal the hidden structure of botanical trees from laser scan data?*

The extracted structure is represented by a skeleton whose extraction process is driven by the following sub-objectives:

1. The extracted skeleton is centered within the represented object and represents the order of object parts correctly, to enable the extraction of tree parameters from point clouds of botanical trees.
2. The skeleton has to be extracted from an unorganized point cloud, which is the given input from a registered laser scan.
3. A skeleton extraction process must be robust to noise, varying point density and undersampling as produced by measuring instruments.

4. Laser scanning produces huge amounts of data. Therefore, the skeletonization algorithm should be fast. The algorithm speed is typically expressed by the computational complexity, which is a machine independent measure of how the algorithm scales in time by increasing the number of input points.
5. The algorithm should contain as few user input parameters as possible.

The possible practical value of this thesis is reflected by three sub-objectives:

1. The applicability of the algorithm should be demonstrated on example trees. If the objectives above are achieved, the practical result is the extraction of branch lengths and diameters along with the branching hierarchy of the tree.
2. Identify current limitations in the application of skeletons due to common practice or dependency on other algorithms.
3. Test the performance on objects other then tree structures, to evaluate the generality of the introduced concepts.

Restrictions applying to this research:

1. The skeleton algorithm is not supposed to operate on other input than point clouds.
2. We consider only scans of leafless trees. Removing leaves from scans in a pre-filtering stage leads to additional occlusion effects, which are considered as data problems within this thesis. Practical examples of removing leaves based on intensity values or geometric properties can be found in [Côtè et al., 2009] and [Dai et al., 2009].

## 1.3 Methodology in a nutshell

The skeleton extraction is based on the idea to investigate the local behavior of a point cloud within a small volume. Such a volume is derived from a spatial subdivision. This behavior is described by an initial graph, which is retracted to the final skeleton by a finite set of rules exploiting the neighborhood relations. This set of rules is preserving the branching order and the key to a low computational complexity.

The skeleton algorithm in this thesis consists of three main modules:

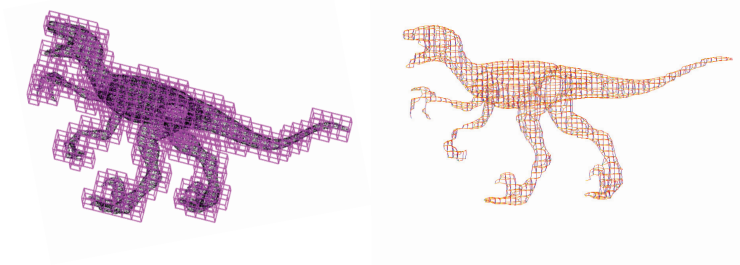
1. **Computation of a labeled octree-graph:** An octree, see Figure 1.9, is built by the subdivision of the point cloud into octree cells. The subdivision is driven by rules based on how the point cloud is crossing through the cell sides. From every octree cell, edges are constructed to neighboring cells, if the point cloud is crossing through the shared cell side as shown in Figure 1.9. The created edges get a triplet as label for the direction, e.g. (-1,0,0) for left. These labels correspond to the principal Cartesian directions. These labels are derived from the octree, which is extracted in the same space as the point cloud

2. **Computation of the Skeleton:** The skeleton is computed (Figure.1.9) by merging two vertices having edges with the same label incident to a common adjacent vertex in the octree graph. Because of the rectangular structure of the extracted octree-graph, this merge of vertices creates a new vertex with 2 incident edges having the same direction label. An edge with the same incident labels is created, if none such edge is present. To achieve this *reduction* a notion of vertex connectivity is used, the so-called *vertex dimension*. The result is the proposed skeleton (Figure.1.9) and called *SkelTre-skeleton*.

3. **Embedding of the Skeleton:** Because of the known correspondence between a vertex and a certain set of points in the original point cloud, the vertices of the skeleton are geometrically embedded. Correspondence is initiated during the construction of the octree-graph and updated during the reduction to the SkelTre-skeleton

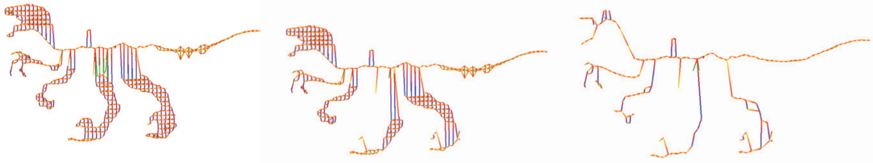
## 1.4 Organization of the thesis

The four main requirements that should be fulfilled by a new skeleton extraction procedure are discussed in Chapter 2. These requirements are describing what a skeleton ideally is and what is required from the extraction process. In Chapter 3 current literature on skeletonization of point clouds is discussed with a strong focus on the skeletonization of trees. For this purpose prior work described in literature is organized into several principal algorithmic groups and their problems are discussed with respect to the background given in Chapter 2. In the core of this thesis, Chapter 4, a new efficient algorithm to reveal the hidden skeleton structure from laser scans, as a skeletonization process is described in detail. The extracted skeleton is called *SkelTre-skeleton* (Skeletonization of Trees). Chapter 5 analyzes the SkelTre-algorithm to verify whether the requirements given in Chapter 2 are met. The practical relevance of such an algorithm is demonstrated on 3D point clouds of dense terrestrial and strongly undersampled airborne laser scanned trees in Chapter 6. In Chapter 6 the extraction of tree parameters by means of the SkelTre algorithm is validated against manual field measurements. Chapter 7 concludes the thesis and gives an outlook to future applications and research remaining on and with skeletons.

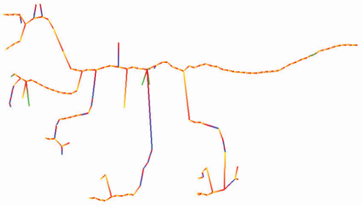


Point cloud
and octree

Octree graph



Reduction



SkelTre-Skeleton

Figure 1.9: The basic steps to compute the skeleton of an object from a point cloud

# 2

# Skeleton extraction requirements

Skeletons are “line-like” representations of objects and give a meaningful order of object parts. Algorithms to compute a skeleton of given objects are under active research. In the last decades hundreds of publications have been published on skeletonization [Biasotti et al., 2008]. Mostly considered are skeletonization algorithms in two- and three-dimensional Euclidean space. Many skeletonization algorithms are developed to operate on two-dimensional data, [Klette et al., 1998], on 3D- polygon models, [Dey and Sun, 2006], or on three-dimensional surfaces, [Biasotti et al., 2008]. Modern measurement instruments produce huge numbers of point measurements, so-called point clouds, and pose algorithmic challenges because of noise, undersampling, varying point density and huge amount of data, [Bucksch et al., 2010]. An overview of skeletonization algorithms for point clouds was published in [Cornea and Min, 2007] and the algorithms were evaluated against 12 properties of a computed skeleton.

Section 2.1 describes the general problem of extracting a skeleton from a point cloud. Section 2.2 introduces four skeleton requirements that should be fulfilled by 1.) a computed skeleton and 2.) by the extraction algorithm. These four requirements are further described in the following pages. Section 2.3 describes which order of object parts we want to preserve and gives a qualitative description of the idea used in the SkelTre algorithm, that will be introduced in Chapter 4. This is followed up by Section 2.4, which describes how the skeleton sizes in the point cloud, independent of the used coordinate system. Section 2.5 discusses the information obtainable from an imperfect point cloud. Section 2.6 introduces a notion to evaluate the efficiency of a skeletonization algorithm.

## 2.1 General problem description

[Cornea and Min, 2007] formulated twelve desirable properties for a skeleton extracted from a point cloud. They distinguished between general and application specific skeleton requirements. Their main conclusion concerning skeletons derived from point clouds was that no skeleton algorithm fulfills all of the desirable requirements stated in [Cornea and Min, 2007]. As a consequence they concluded further that an algorithm has to be designed specifically for each application.

| skeleton | | | | |
|---|---|---|---|---|
| Requirement classes | descriptive requirements | | algorithmic requirements | |
| Properties | topological preservation | metric representation | handling of imperfect data | computational efficiency |
| Attributes | Branching detection<br>Connectedness<br>Thinness | Transformation Invariance<br>Centeredness<br>Smoothness | Operating on point sets<br>Robustness<br>Error model | Efficient<br>Hierarchic |

Table 2.1: Requirements of a skeleton and its extraction process

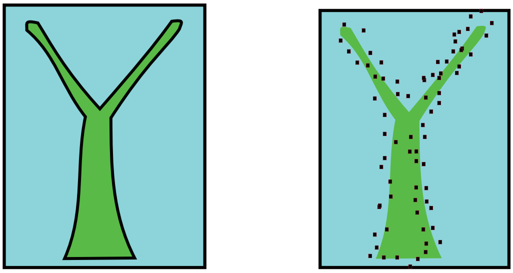


Figure 2.1: (left) The black surface describing an object (right) The same surface, but sampled

For real world objects, we can limit the input for a skeleton extraction to a surface in 3D Euclidean space. Often watertight surfaces are considered, which enclose a volume, as shown in Figure 2.1 on a simplified tree consisting of two branches and a stem. In contrast the extraction of information from data is limited to a sampling of the object surface subject to noise, undersampling and varying point density. We consider here as a sampling the collection of a huge amount of point measurements on a surface. A point cloud of an object is therefore a finite sampling of its surface. Each point of the point cloud is derived from a certain position on the object surface. As it is in fact a measurement, each sampled point is affected by measurement errors. A collection of such points is therefore a so-called *imperfect point cloud* and considered as the given input data from measurement instruments. We denote it mostly as point cloud. In Chapter 1 it was already stated, that we consider point clouds obtained by scanning devices like terrestrial laser scanners. Such a point cloud is shown in principle on the simplified tree in Figure 2.1(right). In general, cavities inside the object are not taken into account, because the measurement instruments considered in Chapter 1 can not capture cavities inside an object, we can only obtain the *outer form* of an object.

One challenge of skeleton extraction methods is to preserve the coherence between the order of object elements and the outer form of an object. As an example: Botanical trees are visually recognized by their object elements stem and branches and their arrangement in space. The branches and the stem are quantified by their size. This

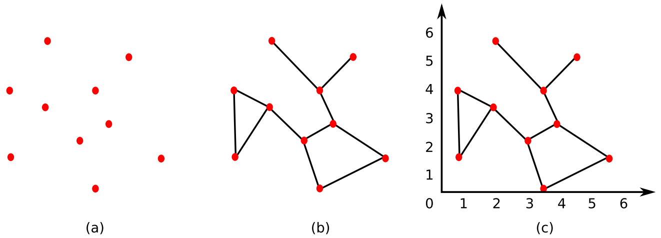


Figure 2.2: (a) a space (b) a topological space and (c) a metric space. (The figure is adapted from [Zomorodian, 2005])

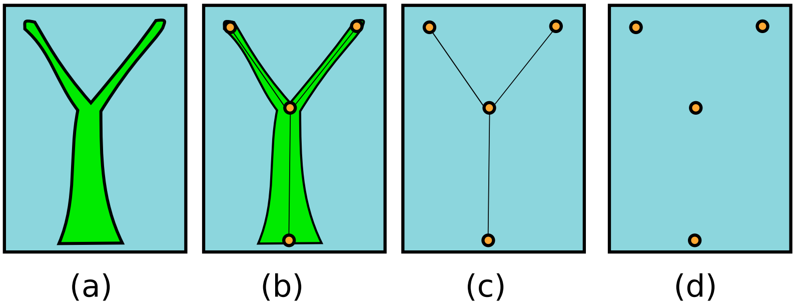


Figure 2.3: (a) a simple tree in 2D with its black boundary, (b) a simple tree with its skeleton, (c) the skeleton of the given tree and (d) the end and branching points of the tree.

size is described by a metric (e.g. Euclidean metric) and expressed using a unit like centimeters. The configuration of elements, e.g. the branches of a tree, is called a topology, [Zomorodian, 2005]. In simplified words, a skeleton enables the answer to the two following questions:

1. How is the outer form defining object parts?
2. How is the size of object parts represented?

We call these two aspects the *descriptive requirements*: *topological preservation* and *metric representation*. Both requirements describe what information is encoded in the skeleton.

Figure 2.2(a) allegorises an arbitrary set of elements. These single elements form a *space*. The conjunction between the elements in Figure 2.2(b) forms a so-called *topological space*, [Zomorodian, 2005], addressing the problem of how the elements are connected. Figure 2.2(c) adds a metric to the topological space, enabling the measurement of a distance between two elements and expression of its location.

In the case of this thesis the object is represented by its *boundary*, Figure 2.3, sampled by a set of point $P$, obtained as single point measurements by typically a laser scanner. Consider the boundary of the given simplified tree in Figure 2.3(a) as a surface. The measurement of branch length and diameter is carried out practically by measuring the size along the external side of a branch. It would be easier, if simply a *line segment* centered inside the branch would be known as shown in Figure 2.3(b). Then the length of a branch corresponds to the length of a line segment and the branch diameter to the double distance of the points on the surface to the line segment. Therefore it is of interest to obtain such a point-line description, a so-called *graph*, [Wilson, 1985], or more precisely a *skeleton graph* as shown in Figure 2.3(c). The desired connections between the points are obtained by connecting the points as shown in, Figure 2.3(c).

Furthermore, one is intuitively locating the branching point like in Figure 2.3(b), where the stem splits into two branches. By connecting the branching points and the end points, Figure 2.3(c) is obtained. This intuitive locating is only possible, because we know the surface of the object from Figure 2.3(a). In fact we can recognize in Figure 2.3(b), that the branching point is in the middle of three locations where the surface changes its direction. The end points of the skeleton graph relate to local extrema of the surface.

Furthermore, two questions are related to the algorithmic challenges of extracting a skeleton. Namely,

1. How is the skeleton extraction addressing imperfections in the data obtained by measuring instruments?

2. How efficient is the skeleton extraction?

The two challenges above lead to the algorithmic requirements of a skeletonization algorithm. These two algorithmic aspects are the requirements *handling of imperfect data* and *computational efficiency*. Both are aiming to how the extraction process of the skeleton is to be designed.

From here we obtain a structured list of two *classes* of *skeleton extraction requirements* (Table 2.1): the descriptive requirements of a skeleton, which represent the information encoded by the skeleton and the *algorithmic requirements*, which represent the requirements of the extraction process from point cloud data. The descriptive requirements are the topological preservation and metric representation. Topological preservation focuses on the links between elements of an object, while the metric representation aims on information encoded by the embedding of the object into 3D Euclidean space. As an example for topology preservation we want to preserve the order of branching and non-branching parts of the object. In this case the branching and non-branching parts are the elements of our topology. An example of metric representation is to enable the extraction of diameters and length of object parts.

The algorithmic requirements are *handling of imperfect data* and *computational efficiency*. The handling of imperfect data describes the limits of the information we can extract from given data. The computational efficiency expresses the speed of the algorithm in extracting the skeleton from the given input. The computational efficiency describes for example the amount of points which can still be handled by a skeletonization algorithm. Each requirement is subdivided into attributes. These attributes can be validated and are ideally observable when the requirement is fulfilled. We are aware that attributes are correlated.

In the following section these requirements are summarized in a requirement list. In the rest of the thesis we focus on finding and validating a construction of a skeleton satisfying these requirements.

## 2.2 Descriptive and algorithmic skeleton requirements

The overall conclusion of [Cornea et al., 2005], was that skeletonization algorithms depend on the application requirements, because no algorithm exists, that fulfills all of their 12 given properties. Therefore, different types of applications need different skeleton extraction methods. We do not consider the properties in the same sense as [Cornea et al., 2005], but we define *requirements*, which we can verify to be met under certain conditions, in the following chapters.

The newly constructed list below contains the two descriptive requirements, topological preservation, metric representation and the two algorithmic requirements handling of imperfect data and computational efficiency, which are investigated in detail in Section 2.3-2.6. The initial list of [Cornea et al., 2005] contained all its properties on an equal level. In our opinion the four requirements topological preservation, metric representation, handling of imperfect data and computational efficiency are the main requirements, further specified by their *attributes*. Here we explain briefly each requirement and its attributes, before we discuss each of the four requirements in the following paragraphs.

### 2.2.1 Descriptive skeleton requirements

In this paragraph a short overview of the descriptive skeleton requirements and their attributes is given.

1. **Topology Preservation:** A skeleton should preserve the links between object elements forming the whole object as represented by the point cloud
    - **Branching Detection:** The branching points of the skeleton should correspond to the parts where the object is branching.
    - **Connectedness:** One skeleton should correspond to one object, such that the skeleton is connected, exactly where the object is also connected.
    - **Thinness:** The skeleton should be one-dimensional, and should not occupy space.
2. **Metric representation:** The skeleton is embedded into $\mathbb{R}^3$ and fits to the surface sampled by the point cloud.
    - **Transformation Invariance:** Every object should have its unique skeleton, which does not change under translation and rotation.

- **Centeredness:** Every part of the skeleton corresponds to a certain subset of the point cloud. The skeleton is centered within the point cloud for each of its corresponding subsets.
- **Smoothness:** The skeleton should be as smooth as the surface of the represented object.

### 2.2.2 Algorithmic skeleton requirements

In this paragraph a short overview of the algorithmic skeleton requirements, considering handling of imperfect data and computational efficiency, and their attributes is given.

1. **Handling of imperfect data** Measurement data may give a non-ideal representation of the object. An algorithm should be as independent from this as possible and address the problems arising adequately.
    - **Operating on Point Sets:** Modern object sampling equipment produces most often point sets as an output. Therefore, the skeletonization algorithm is required to handle point data.
    - **Robustness:** A skeleton algorithm should be robust to noise, undersampling and varying point density.
    - **Error model:** For every kind of data it is in principle possible to derive an individual error description, describing how much a measured point deviates from the original point on the surface. An algorithm should allow to incorporate these error descriptions.
2. **Computational Efficiency:** The efficiency of the algorithm and its capability to operate on huge data sets should be optimized.
    - **Efficient:** Operations on huge datasets, as considered in the applications here, need computation time and occupy computer memory. Both, computation time and memory allocation, should be as small as possible.
    - **Hierarchic:** Finer spatial resolutions should lead to more complex skeletons. As a consequence, roughness of a surface becomes more influential at finer resolutions. The more complex a surface is, the more changes of surface directions will be coded in finer resolutions of the skeleton.

## 2.3 Topology Preservation

A topology contains elements, which are connected by links, as explained in Section 2.1. If a topology is to be preserved in a process, then the elements and links forming the topology have to be specified.

In our case a watertight surface embedded into 3D Euclidean space is the description, of which we want to preserve a topology. This surface is to be separated into parts

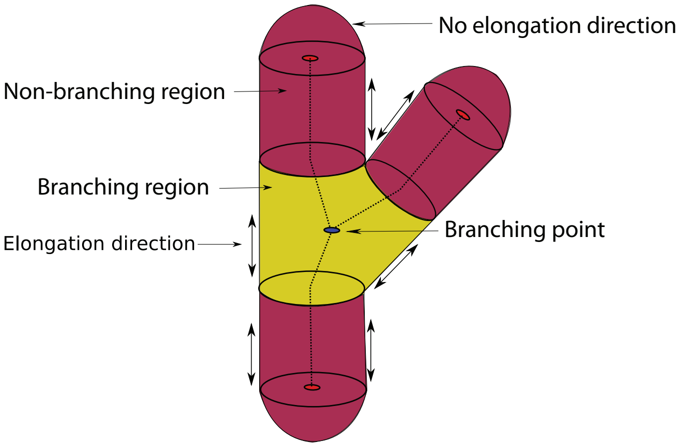


Figure 2.4: The object parts: (reddish) non-branching region and (yellowish) branching regions. The dotted line shows the links between the elements of branching and non-branching regions. The blue point denotes the branching point belonging to the branching region. The red points relate to the non-branching regions. The elongation direction is indicated by double arrows.

corresponding to the elements of the topology. The surface automatically contains the link between separated elements, which fulfills the connectedness attribute.

From the branching detection attribute it directly follows, that it is required to preserve the link between *branching* and *non-branching* regions on the surface. These branching and non-branching regions are the elements of the topology to be preserved:

We assume that a notion of the surface characteristic, the so-called *elongation direction* is given. The elongation direction describes locally in which direction the object surface is sized most. In Figure 2.4 we give one example for the elongation direction. If we shrink the surface in every point in the direction perpendicular to the elongation direction until no shrinking is possible anymore, the result is thin and does not occupy space. Therefore the thinness attribute is met. The shrinked surface contains end points corresponding to those regions where elongation stops to exist.

Branching of the object is indicated by a split in the elongation direction. In such a case, the shrinking stays perpendicular to the elongation direction and branching can occur, when different shrinking directions meet in one point. Such a point is a branching point in the given example and is derived from the surface region information.

In the case considered here we distinguished between two kinds of elements to be linked: *branching regions* and *non-branching regions*. In the example in Figure 2.4 we assume spherical parts of the surface as having no elongation at all. Because of that they are belonging to the non-branching region identified by the elongation direction.

A branching region is represented as an element of the topology having three or more links with other elements. Remember, the links are given by the surface itself. From here we obtain a topology describing the object by means of the elements branching and non-branching region and the links given by the surface. Hence, the next section

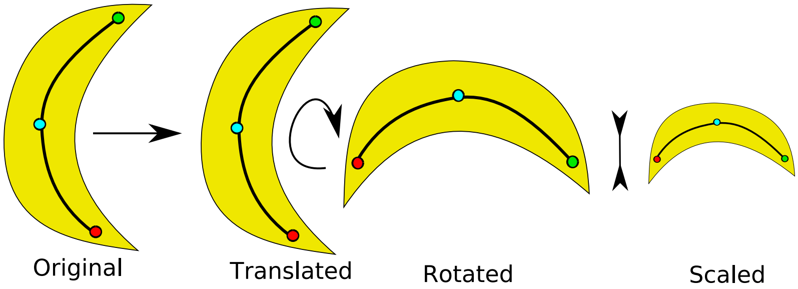


Figure 2.5: Example of the 3 considered transformations on a centered skeleton: Translation, rotation and scaling.

will describe how to embed this topology into a metric space.

**We can conclude from here, that elongation directions allow us to describe branching and non-branching regions of an object.**

## 2.4 Metric representation

As a first attribute of the metric representation we discuss the transformation invariance. This is assured by the relation between the skeleton and the surface. The transformations are shown in Figure 2.5. The transformations are translation, rotation and scaling. Because the elongation direction is describing the surface locally, it transforms in the same way as the surface and is therefore independent from the objects location, orientation and size. Notably, local differences in elongation direction are preserved by the three transformations considered.

Furthermore, we want the skeleton to be centered. Practically this means that we want to represent the links of the previously introduced topology as a curve passing through local centers within the object. If the previously introduced shrinking affects all points on the surface by the same amount until it is thin, than the skeleton is centered. Centeredness is therefore met in the sense, that points on the surface which are shrinked to the same position within the object are perpendicular to the skeleton.

The smoothness requirement is implicitly met, if an object part can be subdivided further in the elongation direction, such that every point on the skeleton corresponds to a unique subset of the surface. The more local the subdivision, the smoother the result is (assuming that the object is smooth on itself).

From the discussion above it follows, that: **the attributes of the metric representation are fulfilled, if we can utilize elongation directions to define the center of an object locally.** It is illustrated in Figure 2.5, that the order of the three points on the black skeleton and the relation between the path length between two parts of those three points does not change under the transformations, translation, rotation and scaling.

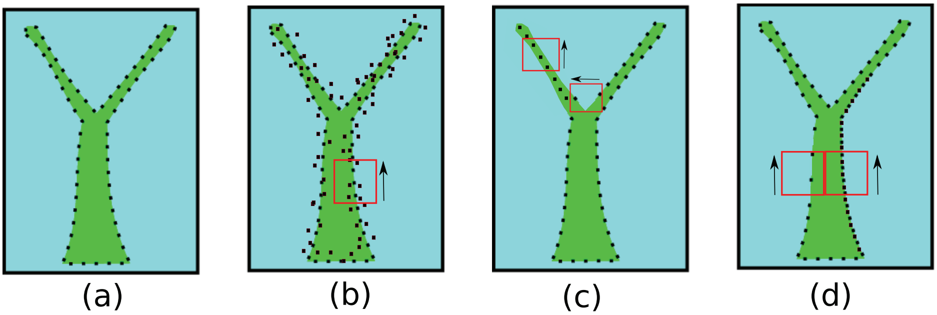


Figure 2.6: (a) An ideal sampling the example surface (black) of the solid tree (green). (b) A possible sampling of the tree containing noise. In the red box it is indicated that even when noise is present a local elongation direction of the surface is extractable and indicated by an arrow. (c) undersampling of an object feature and a possible extractable elongation direction. (d) varying point density and possible extracted elongation directions

## 2.5 Handling of imperfect data

Laser scanners measuring a continuous surface represent the surface as a set of discrete points. Each discrete point is the result of the beam return reflected back on the receiver in the scanner. On the receiver a certain area is covered by the returning beam, from which a distance value is obtained. As stated in Section 2.1, we need to find a link to the descriptive requirements in the input data. In measurement practice the surface is represented by an imperfect point cloud, e.g. [Blais, 2004], [Pfeifer and Briese, 2007].

Three aspects of imperfect data have to be covered by an algorithm operating on it [Bucksch et al., 2009a]:

1. Noise
2. Undersampling
3. Varying point density

In Figure 2.6(a) it is illustrated that the sampling produces points on the object boundary. Unfortunately, real data contains noise [Shan and Todd, 2008], which is “thickening” the surface. We can qualitatively recognize that the point data obtained by a measurement instrument thickens the represented surface to a ’volume-like’ representation, Figure 2.6(b). This problem is for example visible in 1.6 at the end of the branches. Fortunately, even when a point cloud is strongly undersampled close to a “line”, the elongation direction of the surface in a local neighborhood can be identified, as shown in Figure 2.6(c).

Another aspect of real data is undersampling forming holes in the represented surface, which do not belong to the original object. These holes are caused by *occlusion effects* and measurement failures as described e.g. in [Shan and Todd, 2008]. If the surface area around the undersampled surface part is large enough, a rough elongation direction might be found, Figure 2.6(a). Here, occlusion effects are covered as an extreme case of undersampling.

Furthermore varying point density occurs because of mechanical inaccuracies and the different orientation of the scanned object with respect to the scanners spherical scan geometry, [Shan and Todd, 2008]. Varying point density on the surface allows still a proper extraction of the surfaces elongation direction, under the choice of a large enough neighborhood, as shown in Figure 2.6(d). The explanation of imperfect data is rather qualitative, because the exact description of the noise occurring in laser scans is still under active research. Some results on the description of noise on natural surfaces were already obtained by [Hodge, 2010], who mentioned trees explicitly in her experiment.

**We can conclude from this paragraph, that local elongation directions are often extractable from imperfect point cloud data;** because of that local elongation directions are a suitable link between descriptive and algorithmic requirements. Note here a natural link: We used a neighborhood of several points, which allows us to average over a small part of the point cloud to reduce noise, thus enhancing the metric representation requirement. Furthermore, we reduced the amount of used data to calculate the skeleton by grouping points, which leads us to the next paragraph.

## 2.6 Computational efficiency

Modern instruments to produce point cloud data, as the ones used in this thesis, easily capture point clouds in the order of several millions of points. Hence, the efficiency of an algorithm is an important requirement.

A criterion to evaluate the efficiency of an algorithm should be independent from the machine performing the actual calculation. An often used criteria to judge the efficiency of an algorithm is the *computational complexity*.

Computational complexity quantifies the running time behavior of an algorithm, which is some function $f(n)$ of the input size $n$, by the asymptotic upper bound $g(n)$. $g(n)$ tells us that $f(n)$ is up to some constant multiple always less or equal than $g(n)$ (Figure 2.7). Most commonly the *Big-O notation* is used to express the temporal complexity, which is denoted as $O(g(n))$. We consider the order of $g(n)$ as an indicator for the algorithm performance.

**Definition 2.6.1.** *Let n be the length of the input, then* $f(n) = O(g(n))$ *means there are positive constants c and k, such that* $0 \leq f(n) \leq cg(n)$ *for all* $n \geq k$. *The values of c and k must be fixed for the function f and must not depend on n. [Knuth, 1998]*

Let's make a little example to get the importance clear. Suppose you have a point cloud with 100.000 points, which is a realistic amount of points a laser scanner produces in one second. And suppose you design an algorithm to sort this point cloud according to ascending z-values. One way to do this is to simply search first for the smallest value in the list of points, put it to a new list and remove it from the first list. This is repeated until all points are in the new list. This would result in a complexity of

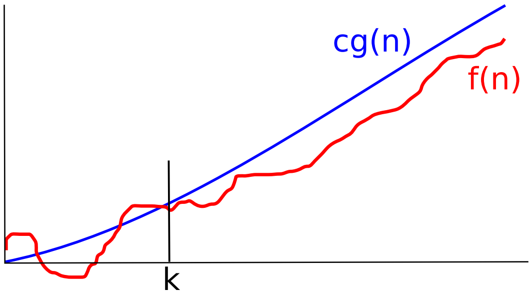


Figure 2.7: The blue graph $g(n)$ is an upper asymptotic bound of the function $f(n)$ up to a constant multiple $k$

$O(n^2)$. If one comparison of two values in the list just takes 1ms, then this results in a computation time of 58 days. Luckily, other sorting mechanisms exist, e.g. merge sort, with a complexity of only $O(nlogn)$. These algorithms make it possible to do the same in maximum 4.2 minutes under the given conditions. In the preceding sections we described already the elongation direction. The changes in elongation direction comprise a fraction of the original input size. Limiting the calculation mainly to such a subset is one way to optimize an algorithm.

An algorithm that can operate on different spatial resolutions, can produce skeletons at different levels of detail. Finer spatial resolutions lead to more local elongation directions that have to be processed. At larger spatial resolution less changes of the elongation direction are addressed. Because of that, a skeletonization algorithm that is able to extract skeletons at different resolution is hierarchical.

Continuing with building up a conclusion, we can state here: **A skeleton is efficiently extractable, if we can extract local elongation directions and the computation is driven only by changes in elongation direction, extractable at different resolutions.**

## 2.7 Summary

In this chapter four general requirements for a skeleton extraction procedure have been presented:

1. Topology preservation
2. Metric representation
3. Reliability
4. Computational Efficiency

It was stated here, that the order of object parts can be preserved in a skeletonization process if information about the local elongation direction is contained in the input data.

The metric representation was described by the properties of the elongation direction, that enable transformation invariance, centeredness and smoothness.

The algorithmic requirements are the efficiency and the handling of imperfect data by the skeleton extraction procedure. We discussed the given input data and showed that in many cases the elongation direction is extractable despite the input data problems noise, undersampling and varying point density. We also introduced the temporal complexity as a measure to evaluate the algorithm performance.

It is concluded from this chapter that potentially all four requirements can be met if an algorithm exists, which utilizes local elongation directions when analyzing the surface. This is assumed to be often extractable from the data and an embedding into local centers of gravity allows to preserve the topology and to represent the skeleton metrically. Dealing efficiently with huge amount of points is done by the use of suitable local centers covering all changes in the elongation direction. The computational complexity is to be optimized purely by the algorithm design. First, in the next chapter, the state of the art literature on skeleton extraction from point clouds is reviewed.

# 3

# Skeleton algorithm classes

In this chapter an overview of existing algorithms for skeletonizing 3D point clouds is given. After an introduction sketching the historical context, two groups of algorithms are described in Section 3.2 and Section 3.3. The first group roughly consists of algorithms developed for 2D skeletonization extended to 3D, while the second group uses a direct approach. This distinction will be more precise in the following pages.

## 3.1 Introduction

The extraction of a skeleton from a point cloud is carried out by a skeletonization algorithm. Until now we defined requirements on a skeleton and its extraction process, building a basis to evaluate current skeletonization algorithms. Only algorithms operating on point cloud data in 3D Euclidean space are of interest in our context, thus this overview is limited to such algorithms. For a more general overview on skeletal structures we refer to [Biasotti et al., 2008] and [Cornea and Min, 2007], which also discussed algorithms for closed surface representations and point clouds in 2D.

From these overviews we can sketch a historical development of skeletonization algorithms. Initially skeletons were computed for objects identified in 2D images. These images have the property, that the complete 2D image space is filled and for every pixel in the image the adjacent pixels are known. The computation of a skeleton relies therefore on an extraction from a watertight object boundary with a defined inside and outside in 2D. Often clustering methods are applied to extract objects from 2D images (e.g. [Lucchese and Mitra, 2001]). Examples for methods originally developed for such extracted 2D objects are morphologic thinning (e.g. [Serra, 1982]) and the distance field (e.g. [Langetepe and Zachmann, 2006]), both extracting the medial axis of an object. Algorithms operating on boundary samples in 2D exist, too. For example the medial axis, which is a skeleton in the 2D case, can be approximated from a set of boundary samples as a subset of the Voronoi diagram (e.g. [O'Rourke, 2005] and [de Berg et al., 2008]). One drawback of the medial axis on imperfect data is its sensitivity to noise, [Cornea et al., 2005].

The extension of these algorithms to 3D laser points clouds representing the boundary

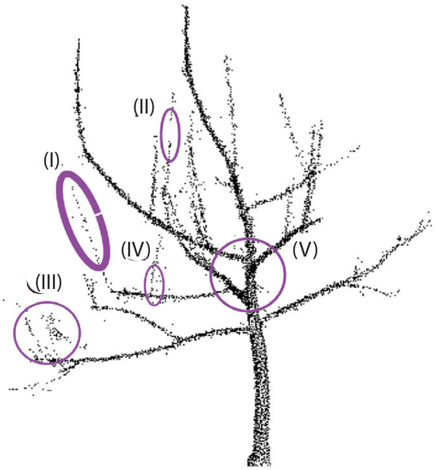


Figure 3.1: Laser scan of the test tree: (I) strongly undersampled region. The branch is already visually not identifiable as a volume, (II) data gap because of occlusion, (III) random noise because of combined effects, (IV) combined occlusion and undersampling and (V) varying point density.

of an object, see also Chapter 2, poses two conceptual problems;

1. The inside and outside of the object has to be defined in a pre-processing step
2. The algorithm results in surfaces, which have to be reduced to a 1D representation in a post-processing step.

In the following an overview of existing algorithm types for point cloud skeletonization is given. Their use on point clouds representing botanical trees is highlighted, because trees are our target application. An overview of algorithm types is given in Table 3.1. We distinguish algorithms based on the dimension of the output descriptor. Two classes are shown in Table 3.1, which we discriminated by their descriptor dimension. First, algorithms extracting a 2D descriptor, which is reduced further to a one-dimensional skeleton and secondly, algorithms producing a one-dimensional skeleton directly. Further distinction is achieved by identifying the underlying data structure and the computational complexity of the algorithms, which is an important factor when using large data sets.

The algorithms are demonstrated, if available, on a test tree (Figure 3.1). This test tree contains typical problems of laser scanned point cloud data: These problems are undersampling, noise and varying point density and exist both in registered and unregistered point clouds. The scan of the test tree contains 7183 points and was done with a Zoller + Frölich Imager 5003 Scanner on an apple orchard near Kentville, Nova Scotia, Canada.

| **Algorithm class** | **Descriptor dimension** | **Spatial data structure** | **Complexity** |
|---|---|---|---|
| Morphology | 2D | Raster with defined boundary | $O(nw)$,[1,2] |
| Distance transform | 2D | Raster with defined boundary | $O(n)$,[1] |
| Voronoi diagram | 2D | Defined boundary | $O(n^2)$,[1,3] |
| Clustering | 1D | Neighboring structure, e.g. kd-tree or minimum spanning tree | $O(kn^2\Delta^2)$,[1,4,5] |
| Level set extraction | 1D | Raster with defined boundary | $O(n)$,[1,6] |
| Graph reduction | 1D | Octree graph | $O(n)$,[1] |

Table 3.1: Algorithm classes. [1]$n$ denotes the number of input cells, point cloud points or vertices, [2]$w$ denotes the size of a structuring element, [3]worst case scenario, often almost linear in practice, [Amenta et al., 2001],[4]guaranteed scenario for $k-means$ clustering [Arthur and Vassilvitskii, 2005], [5]$\Delta$ is the spreading of the input points, [6]referring to the height function, may be higher for other functions.

## 3.2 2D Skeleton

The best known 2D descriptor of an object is the medial axis, Figure 3.2, which is the set of points having more than one closest point on the object boundary, [Blum, 2007]. Note, that the medial axis is also a formulation of centeredness and several frameworks exist to extract a medial axis. The medial axis may be derived as a subset of the Voronoi diagram of the point cloud, [O'Rourke, 2005] and [de Berg et al., 2008], or from a morphological thinning process, [Serra, 1982]. One major property of the medial axis of a 3D object is that it is not necessarily one-dimensional, but in general consists of a set of surfaces and lines in 3D. For the reduction of the 2D medial axis to a 1D skeleton, a second processing step is needed. For example, [Dey and Sun, 2006] have shown that an approximate medial axis is reducible to a meaningful skeleton. Medial axis approaches commonly need to define inside and outside of the object.

### 3.2.1 Morphological thinning

Morphological thinning methods organize the point cloud in a 3D raster of equally sized cubic cells. From this raster the outer layer is removed until the skeleton remains. The outer layers are removed using the morphological operations opening and erosion, [Serra, 1982], as explained in more detail below. This class of algorithms requires a defined inner volume of the object to produce a centered skeleton. Depending on the kernel used for the thinning process also the medial axis is derivable. [Palagyi

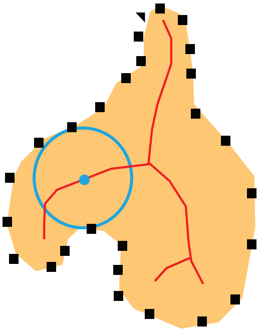


Figure 3.2: Example of a medial axis: (red) the medial axis. (blue) a point on the medial axis and its closest points on the boundary of the largest empty circle, (black) the point cloud sampling of the surface. The picture is redrawn from [Amenta et al., 2001].

et al., 2001] introduced a time linear algorithm using 6 sub-iterations. Its application on tree point clouds was proposed by [Gorte and Pfeifer, 2004] and later extended by [Gorte, 2006]. As stated by the authors, the resulting morphological skeleton does not preserve the object topology, [Gorte and Pfeifer, 2004]. Varying point density is treated by taking into account the number of points per cell. Undersampling and occlusions are treated by using an user defined octree cell size. The efficiency of morphological thinning is characterized by a complexity of $O(nw)$, $n$ being the number of raster cells and $w$ being the number of cells used as a structuring element. The following explanations of morphology can be found in e.g. [Serra, 1982].

The theory of morphology is based on two complementary operations on sets. Let $A$ be the set of raster cells, e.g. containing point cloud points, which is modified with a set $B$, the so-called *structuring element*.

The two basic operations *erosion* and *dilatation* are defined as:

**Definition 3.2.1.** *Erosion:* $A \oplus B = \{x | B_x \cap A \neq \emptyset\}$*;*
*Dilatation:* $A \ominus B = \{x | B_x \subseteq A\}$*. [Serra, 1982]*

Here $B_x$ denotes the translation of $B$ in direction of vector $x$.

For thinning, a combined operation, called *hit-and-miss-transform*[1] is used to separately treat the solid and the surrounding void. That is, the hit-and-miss-transform is the combination of two structuring elements: $B_B$ for the surrounding void, so-called background, and $B_F$ for the voxelized solid, so-called foreground. The intersection of the eroded foreground with the eroded background is called hit-and-miss-transform:

**Definition 3.2.2.** *Hit-and-miss-transform:* $A \otimes B_{F,B} = (A \ominus B_F) \cap (A \ominus B_B)$

A *morphological thinning* is defined as the intersection of $A$ with the result of the hit-and-miss-transform of $A$, [Serra, 1982].

---

[1] Also the term Hit-or-miss transform is used in literature and refers to the same transformation

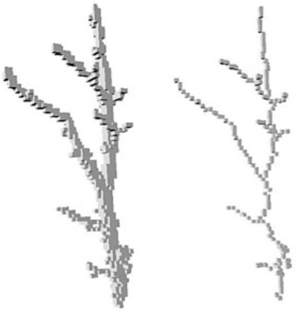


Figure 3.3: Skeleton extracted by morphological thinning with the algorithm of [Gorte and Pfeifer, 2004]. (Left) non-empty voxels (right) extracted skeleton

**Definition 3.2.3.** *Morphological thinning: Let A be a set of voxels marked as foreground and background voxels, then the thinning of A is given by* $A_{thin} = A \cap (A \otimes B_{F,B})$

In Figure 3.3 a morphological thinning procedure is illustrated. On the left the input voxels are shown. These are the voxels containing at least an user defined amount of point cloud points. On the right the extracted skeleton is shown.

If the volume of the original real object can be correctly filled with voxels, the skeleton can be centered and is thin. But the problems with data gaps and additional noise specified in Section 2.5 make this a difficult and to my best knowledge an unsolved task. In Figure 3.4 the result of the morphological thinning method on the example tree of Figure 3.1 is shown. That the skeleton is not always centered is visible in Figure 3.4 from the waviness of the branches, resulting from noise and undersampling. It is remarkable that a relatively high resolution of the raster (1cm) and a structuring element of 7 times the raster size was needed to produce the skeleton. In comparison we will see in Chapter 4 that comparable good skeletons were extracted from “raster resolutions” of 10 cm with the skeleton algorithm developed in this thesis.

### 3.2.2 Distance Transform

In practice *distance transform* based methods often start from a point cloud embedded in a 3D raster of equally sized cubic cells, [Zhou et al., 1998]. All raster cells are consecutively marked by their distance to the object boundary. The set of neighboring cells, where distances to the boundary are largest, form the skeleton. These methods extract the medial axis, and face the same post processing problems as morphological thinning approaches. Furthermore, connectedness of the skeleton is not guaranteed [Cornea and Min, 2007]. The computational complexity of the distance transform is given as $O(n)$, with $n$ being the number of raster cells.

The distance transform has only been applied to trees on a small example in [Cornea and Min, 2007]. Of all methods investigated in [Cornea and Min, 2007] the distance

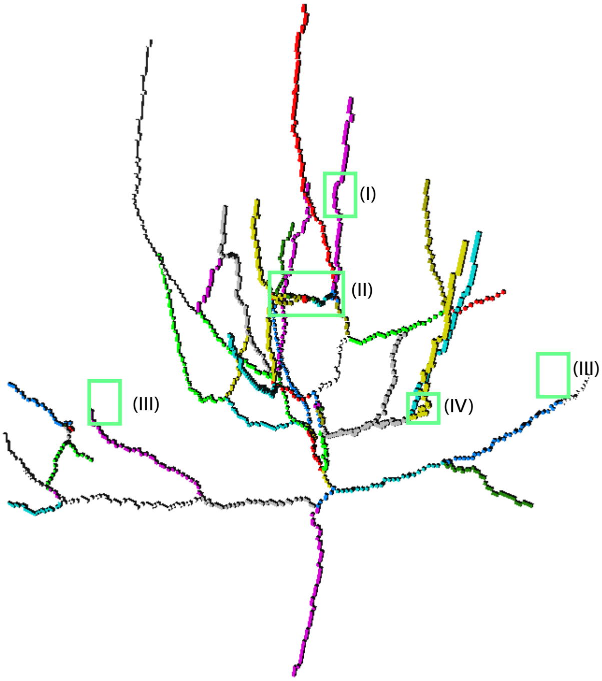


Figure 3.4: Skeleton extracted for the test tree by morphological thinning with the algorithm of [Gorte and Pfeifer, 2004]. The skeleton was extracted from a 1cm raster using a sphere of 7 cm diameter to dilatate the initial raster. (I) shows a location of insufficient centeredness, (II) shows a wrong connection in the skeleton and (III) shows a missing branch and (IV) shows a small part of the skeleton which forms a surface area indicting a violation of the desired thinness of a skeleton.

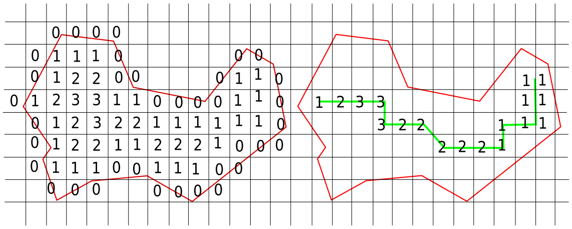


Figure 3.5: (a) The complete distance field of a 2D object defined by its bounding surface (red) (b) the skeleton from the extracted maxima

transform showed the worst performance with respect to the given requirements in Chapter 2. The method is given here for completeness, because it was one of the first popular methods to extract a skeleton.

### 3.2.3 Voronoi Diagram methods

The idea behind *Voronoi Diagram* methods is to extract first a medial axis from the point cloud, and to reduce it further to a one-dimensional skeleton in a second step. Here we briefly discuss one possible combination of two methods. First we extract the medial axis as described in [Amenta et al., 2001] and secondly we reduce the medial axis with the method of [Dey and Sun, 2006]. A similar combination can be found in [Cornea and Min, 2007].

Considering a set of points $P$ in $\mathbb{R}^3$, a Voronoi cell $V_p$ for each point $p \in P$ is defined as the set of all points $x \in \mathbb{R}^3$ that have no other point in $P$ closer to it than $p$. The Voronoi diagram $Vor(P)$ is the complex of all Voronoi cells. Each Voronoi cell in three dimensions has faces of different dimensions. The Voronoi cells, the Voronoi facets, the Voronoi edges and the Voronoi vertices. Note here, that these Voronoi cells are three-dimensional polytopes some of which are unbounded. This paragraph borrowed heavily from [Dey, 2007].

The medial axis is extracted by using the so-called *poles* of the Voronoi diagram of a point cloud. The poles are a subset of the Voronoi vertices. Each point $p$ defines 2 poles. An outer pole $q^+$ of a point $p$ is the Voronoi vertex $q$ of the Voronoi diagram $Vor(P)$ most far away from $p$. Note that $q^+$ is at infinite distance if $p$ is part of the convex hull of $P$. The inner pole $q^-$ is the Voronoi vertex $q$ of $Vor(p)$ farthest from $p$ such that the angle between the two vectors $pq^+$ and $pq^-$ is greater than $\frac{\pi}{2}$, [Dey, 2007]. In the *power crust* algorithm, [Amenta et al., 2001], all poles are distinguished as inner and outer poles (Figure 3.6) of a suitable distance weighted Voronoi diagram, the so-called *power diagram*. The set of inner poles is taken as a good approximation of the medial axis, [Amenta et al., 2001]. The output of the power crust algorithm is a set of polygons describing the medial axis, which is a surface in 3D. The reduction of the medial axis to the skeleton was carried out with the method described in [Dey and Sun, 2006], who eroded the extracted medial axis based on a distance field with the geodesic distance on the medial axis as a metric.

To my best knowledge no specific application of a Voronoi Diagram method to botanical trees is described in literature. This strategy is mentioned in [Cornea and Min, 2007] but not shown explicitly. As stated in [Amenta et al., 2001], this method requires a sufficiently dense sampled object as input, as the object is assumed to be watertight. This condition may be difficult to achieve with laser scanned data on trees suffering from large undersampling. The construction of the Voronoi diagram determines the efficiency of the algorithm. The complexity is in the worst case $O(n^2)$ with $n$ being the number of point cloud points. It was stated in [Amenta et al., 2001] that the complexity is $O(n \log n)$ on average, dependent on the input data.

As an example for a Voronoi based method, the skeleton was extracted of the test tree (Figure 3.1). In Figure 3.7, the result of extracting the medial axis with the approxima-

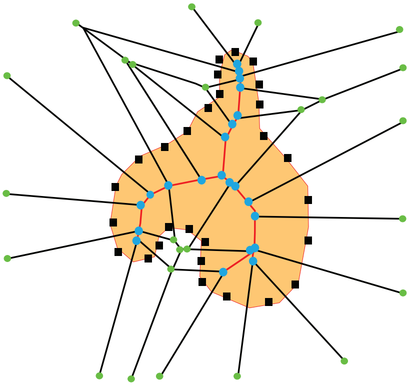


Figure 3.6: A point cloud (black) and the corresponding Voronoi Diagram (black lines) and the medial axis (red lines), the inner poles (blue) and the outer poles (green). The picture is redrawn from [Amenta et al., 2001].

tion of the power crust algorithm and further reduction with the method of [Dey and Sun, 2006] is illustrated. Because the result is heavily distorted and not representing the tree structure, I want to state explicitly, that this does not reduce the value of the work of [Dey and Sun, 2006] and [Amenta et al., 2001], as they had not the application given in this thesis in mind. The reason of this extreme result is twofold. First the sensitivity of the medial axis to noise, and second the extraction of the medial axis needs sufficiently high sampling, as stated in [Amenta et al., 2001].

## 3.3 1D Skeleton

One-dimensional descriptors have in common, that they use neighborhood information to extract the skeleton as a graph. This graph is embedded with an embedding strategy in the point cloud.

### 3.3.1 Clustering Methods

Clustering methods produce clusters of point cloud points from a spanning graph, like the minimum spanning tree, [Meyers et al., 1992], to represent a point neighborhood. Such a spanning graph connects all points of the point cloud. Some distance metric is used to produce the clusters of neighboring point cloud points. Neighboring clusters are connected to a skeleton. In [Xu et al., 2007] a neighboring graph is used with a defined root point being the graph vertex with the lowest z-coordinate. Points with the same distance from the root point are considered as belonging to one cluster. The approach showed good results until two-third of the tree height on a test tree with leaves. The remaining skeleton is generated by using species dependent allometric relationships. Another promising clustering approach was presented by [Yan et al., 2009]. A comparison of the two mentioned clustering methods is illustrated in Figure 3.8. They

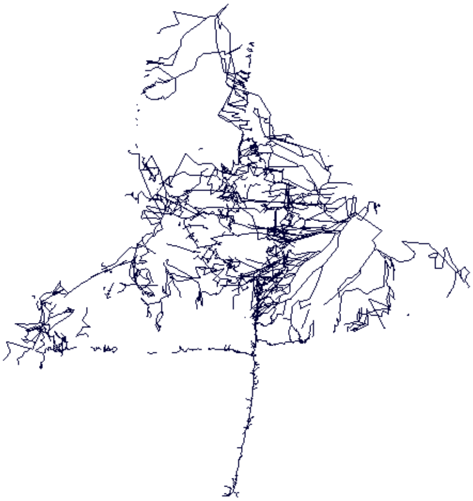


Figure 3.7: Example of a Voronoi method: the medial axis was extracted with the power crust algorithm, [Amenta et al., 2001], and the medial axis was further reduced to the skeleton with the method of [Dey and Sun, 2006].

used a $kd-$tree and $k-$means clustering to produce the clusters from which the skeleton is derived.

It was shown in [Yan et al., 2009] that embedding the skeleton is still an issue for these methods as shown in Figure 3.9. A further drawback is that they operate on all point cloud points to perform the skeletonization. The complexity of k-means clustering algorithms is given by [Arthur and Vassilvitskii, 2005] as $O(kn^2\Delta^2)$, with $\Delta$ being a value representing the spreading of the data. Nevertheless they also showed that the complexity can be reduced depending on the data.

### 3.3.2 Level set extraction

One dimensional descriptors to describe objects with proved topological properties, like the *Reeb-graph* [Reeb, 1946], were used first in [Shinagawa et al., 1991] in a shape retrieval application. Two frameworks exist to extract a skeleton based on topological properties: Extracting the level sets and graph reduction.

Level set extraction methods use the evolution of a piecewise linear function on the sampled surface. On the basis of this chosen function a set of contours is extracted. Given a piecewise linear function $f$, the level set of a value $s \in \mathbb{R}$ is defined as the set of points with a function value equal to $s \in \mathbb{R}$. The construction of a skeleton is based on the analysis of the evolution of the connected components of the level sets generated by $f$. We call such a level set to be merged a *contour*. The principle of level set extraction is shown in Figure 3.10. The height function $h(x) = z$ is often used to extract the level sets from a point cloud as shown in Figure 3.10, e.g. [Verroust and

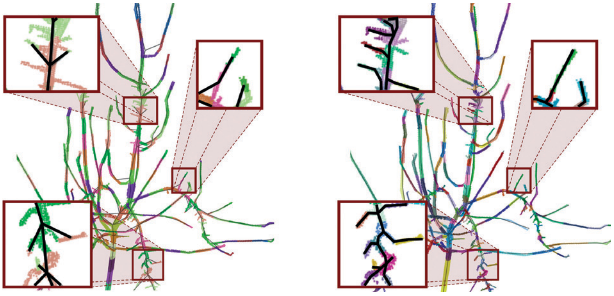


Figure 3.8: Comparison between two clustering methods. The background represent the clustered point data overlaid with the skeleton. Left: Skeleton generated using method of [Xu et al., 2007] with 357 clusters. Right: skeleton using the method of [Yan et al., 2009] with 356 clusters. The figure is taken from [Yan et al., 2009].

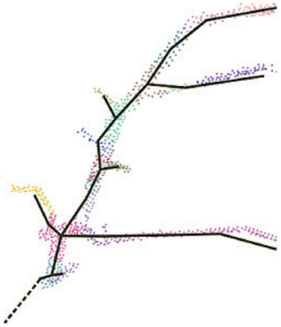


Figure 3.9: Problems with the graph embedding: The skeletal line is not centered everywhere within the point cloud. The figure is taken from [Yan et al., 2009].

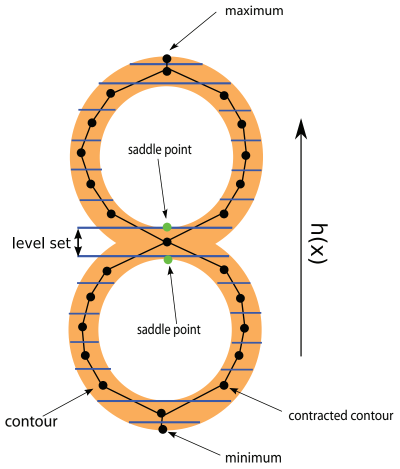
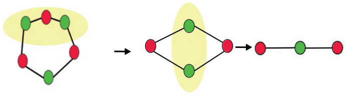


Figure 3.10: The evolution of the height function $h(x)$ over a double torus. The resulting graph with its vertices placed in the center of the extracted level sets. The maximum, minimum and the saddle points extracted by the height function are indicated. Note that under the chosen evolution steps the saddle points are represented as one branching point in the graph.

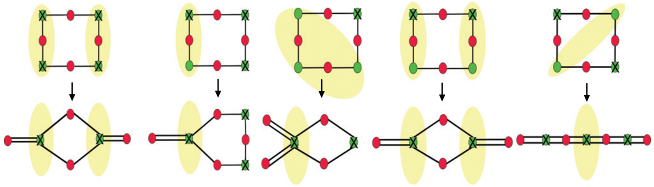
Lazarus, 2000] and [Attene et al., 2003]. Placement of vertices at every centroid of each extracted level set produces a skeleton by connecting the vertices with respect to the chosen function ( see also Section 2.3). Every branching point of the resulting skeleton is a saddle point and every one-connected vertex is a minimum or maximum. The graph containing only the saddle points and the minima and maxima is referred to as a Reeb-graph [Cole-McLaughlin et al., 2004]. The biggest problems arising with these approaches are the rotational dependency of the height function and the sensitivity of the level set extraction to the sampling density, [Cornea and Min, 2007], as already discussed in Section 2.5. The approach of [Verroust and Lazarus, 2000] was applied by [Côtè et al., 2009] on trees. A virtual tree growth model was used based on allometric relationships of known tree species to reconstruct the finer branches. The virtual tree model is required because of undersampling of the finer branches.

### 3.3.3 Graph reduction

A *graph reduction* approach extracts an initial graph from a spatial subdivision using e.g. an octree, [Bucksch and Lindenbergh, 2008]. This initial graph is reduced by a set of rules, Figure 3.11, to a skeleton. These rules consider the connectivity between different parts of the point cloud. Several advantages of such an approach could be demonstrated: a high robustness to noise on imperfect data, a good centeredness and a good connectivity. Centeredness is achieved by embedding the graph into the point cloud. Topological correctness is achieved by choosing a proper decision criterion to place connections between the different point cloud parts and the careful design of the reduction rules. [Bucksch and Lindenbergh, 2008] used two different kinds of vertices,



Basic rule



Reduction cases leading to the basic rule

Figure 3.11: The basic rule is shown on the top, which reduces locally a basic configuration to a line. The different cases occurring are reduced to the basic configuration, which is reduced further to a line. The green vertices are the M-vertices and the red vertices represent the T-Vertices. The M-Vertices with more than two incident edges are marked. The figure is taken from [Bucksch and Lindenbergh, 2008]

so-called M-Vertices and T-Vertices, where M indicates the center of gravity of the point clouds points within an octree cell and T the point where an octree cell side is crossed. This initial graph is reduced by a set of rules, based on the connectivity of the M-vertices, as shown in Figure 3.11. All possible combinations of M- and T-vertices are reducible to a minimal configuration given as basic rule in Figure 3.11. From there the reduction to a skeleton line is guaranteed.

A benefit of their method is that no preprocessing is necessary to define the inner and outer side of the given object. Problems arise with the method of [Bucksch and Lindenbergh, 2008], because noise, undersampling and varying point density are only treated by adapting the local cell size. In case of large undersampling, erroneous loops may appear at higher resolutions, because strongly undersampled parts are treated as holes in the surface. As a consequence the topology of the object is locally not represented correctly. Another aspect on this particular skeleton is that local surface directions were not taken into account, which demanded high measurement resolutions to compute a smooth skeleton. The overall complexity is $O(n)$, with $n$ being the number of point cloud points. The complexity of the reduction with respect to the vertices and edges of the graph to be reduced was not given. An extraction of the skeleton from the test tree in Figure 3.12 shows the rough resolution of the so-called *CAMPINO* skeleton (**C**ollapsing **a**nd **m**erging **p**rocedures **in** **o**ctree-graphs). The skeleton shown in Figure 3.12 will later come back and be compared to the skeleton obtained with a new strategy for graph reduction.

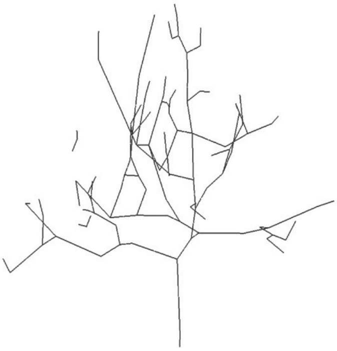


Figure 3.12: Example of an extracted CAMPINO Skeleton. The skeleton shows clearly a loop and some problems in the order of the branches.

## 3.4 Summary

Six different types of algorithms were introduced in this chapter. They were discriminated by the dimension of their output descriptor. We have shown the problems of existing algorithms with respect to the requirements given in Chapter 2. To summarize: none of the existing algorithms meets all requirements. All extensions of existing algorithms require either pre- or post-processing as a separate methodology.

Furthermore, all of the algorithms either operate directly on the whole point cloud or reduce the point cloud first by voxelizing (reduction of the resolution). We conclude that the reviewed algorithms are currently hampered by the following restrictions:

1. inability to preserve the topology of branching and non-branching parts, because the extraction mechanism introduces no difference between branching and non-branching parts, e.g. level set methods, clustering
2. idealization of the object volume, e.g. morphologic thinning and distance transform causing insufficient metric representation
3. insufficient handling of imperfect data causing hardly interpretable results, e.g. Voronoi methods
4. computational inefficiency by operating on the whole input point set, e.g. Voronoi based methods, clustering methods, level set methods

The examples given in this list refer to the methods where restrictions apply most.

As this thesis aims at the extraction of a skeleton-graph, later on called SkelTre-Skeleton, the graph reduction is the chosen principle to be developed further. In the following chapter the descriptive requirements of a new skeleton algorithm will be introduced and the direct extraction of a one-dimensional skeleton-graph will be explained.

# 4

# SkelTre Skeletonization

In Chapter 3 several *algorithm classes* were described to review the principal skeletonization algorithms. This chapter describes a new graph-reduction algorithm developed for the skeletonisation of a laser scanner point cloud. Scanner-produced point cloud data are not only subject to noise, but also to undersampling and varying point density, making it challenging to extract a topologically correct skeleton, as explained in Chapter 2.

In Chapter 2 we have already seen that topological preservation and metric representation of the skeleton are two separate aspects. Also in the algorithm we will find back this separation, the object topology is preserved by a suitable local graph reduction, and the metric representation is achieved by placing the vertices of the resulting skeleton graph in the source point cloud with a suitable graph embedding.

Five sections build the structure of this chapter. First an overview of the algorithm idea is given in Section 4.1, secondly the octree graph extraction is described in Section 4.2. The *computational framework* of the graph reduction is introduced in Section 4.3 to prove, that the SkelTre algorithm preserves the topology of the object sampled by a point cloud. Section 4.4 is describing the graph embedding to achieve a valid metric representation and is followed up by Section 4.5, that summarizes the highlights of this chapter. This chapter is based on the articles [Bucksch et al., 2009a] and [Bucksch et al., 2010].

## 4.1 Algorithm in a nutshell

The skeletonization introduced in this chapter is called *SkelTre*, as it was originally developed for **skel**etonization of **tre**es.

The input of the algorithm is an unorganized point cloud. From here, the point cloud is divided into subsets by an octree subdivision. An octree subdivision is a hierarchical division of a space into cubical cells, so-called *octree cell*s. The centroids of the point cloud points within an octree cell, are the vertices of the octree-graph. Two centroids are connected by an edge if the points of the two adjacent octree cells fulfill a suitable robustness criterion. These connections between centroids are the edges of the octree-graph and get labeled by a direction label to indicate the local elongation of the object surface.

The derived skeleton-graph is based on a well known topological structure, the so-called Reeb-graph, [Reeb, 1946]. It is shown that the reduction of the octree-graph to a skeleton graph by using suitable vertex pairs, so-called *E-Pairs* and *V-Pairs*, embeds the SkelTre skeleton into the Reeb-graph framework. This construction ensures that the derived skeleton is suitable for shape analysis.

To ensure that the derived skeleton also metrically represents the sampled surface, a suitable *graph embedding* is introduced. The *reliability* of this skeletonization method against noise, undersampling and varying point density of the point cloud is ensured by a *robustness criterion*, which we introduce.

The three principal steps of the SkelTre algorithm are:

1. Extraction of a so-called *octree-graph* from an *octree* organization (Section 4.2),
2. Reduction of the octree-graph to a *skeleton-graph* (Section 4.3) and
3. Embedding of the skeleton graph into the point cloud (Section 4.4).

## 4.2 Octree-graph extraction

This section describes the octree-graph extraction procedure.The octree-graph extraction is a two step procedure. First an octree is generated, followed by the robust extraction of an octree-graph along with a *labeling* technique. The motivation to use an octree is to enable local elongation analysis, while still not needing to process all individual points.

### 4.2.1 Octree generation

An *octree* is a hierarchical subdivision of a starting cube into 8 equally sized subcubes, which are the octree cells. These subcubes are subdivided further until the subdivision is terminated.

Let the surface of an object be represented by a point cloud $\Sigma$. A spatial subdivision of $\Sigma$ into subsets $\Sigma_i$ is obtained by an octree.

**Definition 4.2.1.** *The octree space is modeled as a cubical region consisting of* $2^n \times 2^n \times 2^n$ *unit cubes, where n is the subdivision depth. Each unit cube has value 0 or 1, depending on whether it contains data points or not. Adapted from [Chen and Huang, 1988].*

Note that this definition of the octree space assumes equal octree space depth.

Ideally our intermediate result at this stage is a subdivision which separates all parts of the object that are also spatially separated. Clearly, the separating power of the subdivision depends on the minimum resolution of the octree.

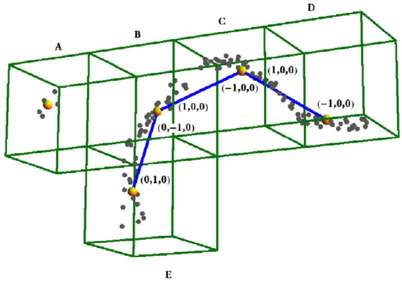


Figure 4.1: Connecting and labeling an octree-graph: Five complete octree cells are shown, containing black data points. The vertices of the octree-graph corresponding to the octree cells are shown in orange. They are positioned in the center of gravity of the local point cloud points.

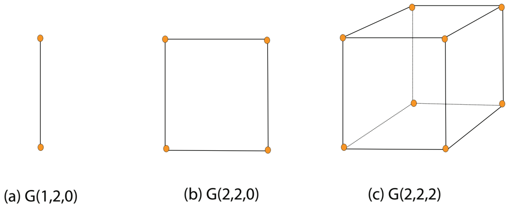


Figure 4.2: (a) a G(1,2,0) subgraph, (b) a G(2,2,0) subgraph and (c) a G(2,2,2) subgraph

### 4.2.2 Extraction and labeling of the octree-graph

We are aiming at a graph-reduction method (see Chapter 3), hence an initial graph, so-called *octree-graph*, is generated. This octree-graph is later reduced to the SkeTre skeleton. An octree-graph is the dual of the octree space, whose vertices correspond to octree cells. The vertices of the octree-graph are simply placed at the center of gravity of all points belonging to a cell, Figure 4.1, and connected by an edge if two octree cells have adjacent faces.

**Definition 4.2.2.** *Let $OC_i, i = 1..n$ be a collection of octree cells. And let $CS_{jk}$, for some $j \neq k, ; j, k \leq n$ be the adjacent sides of each pair of octree cells. The octree-graph $OG(V, E)$ contains the vertices $V$ dual to $OC_i$ connected via the edges $E$ dual to $CS_{jk}$.*

The benefit of using this dual between an octree space and an octree-graph is that it exhibits local grid graph properties:

**Definition 4.2.3.** *A three-dimensional grid graph is an $m \times n \times r$ graph that is the graph Cartesian product of path graphs on $m$, $n$ and $r$ vertices [Pemmaraju and Skiena, 2003]. The grid graph is denoted as $G(m, n, r)$.*

The essentially different locally possible grid graphs in our 3D case are illustrated in Figure 4.2. Here a G(2,2,2) grid graph indicates the unit edges of a full cube. If, say,

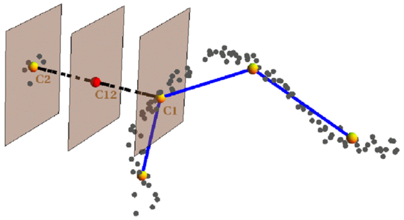


Figure 4.3: Robustness criterion to connect octree-graph vertices $C_1$ and $C_2$ by an edge

z=0, we obtain G(2,2,0) which indicates the unit edges of a square perpendicular to the z-axis. If we change y=2 into y=1, we obtain G(2,1,0) which indicates a unit edge in this square perpendicular to the y-axis.‘

Ideally, the octree-graph represents the local directions of the object surface. As stated above, terrestrial laser scan data is subject to noise, undersampling and varying point density. This can lead to edges in the octree-graph, representing a connection between object parts, which are not connected in the original object. Already in Figure 1.6 and Figure 3.1 typical problems arising with scanner obtained data were shown. Problems in the data can result in both overconnecting and underconnecting. Overconnecting occurs when additional erroneous connections are created due to noise and blunders, underconnecting occurs because of undersampling due to occlusions.

### Robustness criterion

The robustness criterion removes edges from the octree graph, which are wrongly inserted because of data characteristics such as noise and is mainly motivated by two aspects. Firstly, to address the problem of noise, which covers and hides the underlying object topology. And secondly, the robustness criterion can be seen as a module in the SkelTre algorithm, which allows a future extension of the algorithm with instrument specific error models. These two motivations follow directly from Chapter 2.

This robust criterion whether to connect two octree-graph vertices by an edge corresponding to two adjacent octree cells is based on the distances of the cell points to three suitable planes. In Figure 4.3 we apply the robustness criterion to the point cloud and its corresponding octree cells in Figure 4.1. Let $C_1$ and $C_2$ be the centroids of the point cloud points in two adjacent octree cells $OC_1$ and $OC_2$. Let $C_{12}$ be the midpoint of the line segment $\overline{C_1C_2}$. The three suitable planes $P_1$, $P_2$ and $P_{12}$, are the parallel planes through the points $C_1$, $C_2$ and $C_{12}$, perpendicular to the line through $C_1$ and $C_2$. Let $d_1$, $d_2$ and $d_{12}$ be the median values of the squared distances of the points in $OC_1$, $OC_2$ and $OC_1 \cup OC_2$ to the planes $P_1$, $P_2$ and $P_{12}$ respectively. Under ideal conditions the $\sqrt{d_{1,2}}$ of two connected cells would be at least $\frac{1}{4}$ of the distance between $C_1$ and $C_2$ to indicate a connection between the two corresponding point cloud parts. In that sense we use $\frac{1}{16}d_{12} \leq \min(d_1, d_2)$ as a criterion to place connections in the octree-graph.

### Labeling of local directions

In this paragraph we describe how labels are added to the edges of the octree graph. These labels describe the directions with respect to a Cartesian coordinate system where the original point cloud is embedded in. The direction labels will be used to identify the local object elongation which in turn enables a local reduction of the octree-graph. The major benefit of this localized approach is that it overcomes the rotational dependency (Section 2.2) of the level sets $s_z$ of a global height function $f : \mathbb{R}^2 \to \mathbb{R}$. Therefore, we consider level sets locally with respect to one of the three Cartesian directions. That is, dependent on the local object elongation, the height change in either the $x-$, the $y-$ or the $z-$direction is considered. This choice stems from the fact that parts of the surface parallel to the $x-$ or $y-$axis will collapse to one remaining vertex in the skeleton graph, when taking only a global height function into account. Thus, a local choice of $f$ based on the surface elongation will preserve the representation of surface parts parallel to one of the axes.

**Definition 4.2.4.** *A label associated with an edge of the octree-graph indicates the direction of the edge with a direction vector. The labels for all 3D-directions are:*
*Right/Left:* $(\pm 1, 0, 0)$,
*Up/Down:* $(0, \pm 1, 0)$,
*Front/Back:* $(0, 0, \pm 1)$.

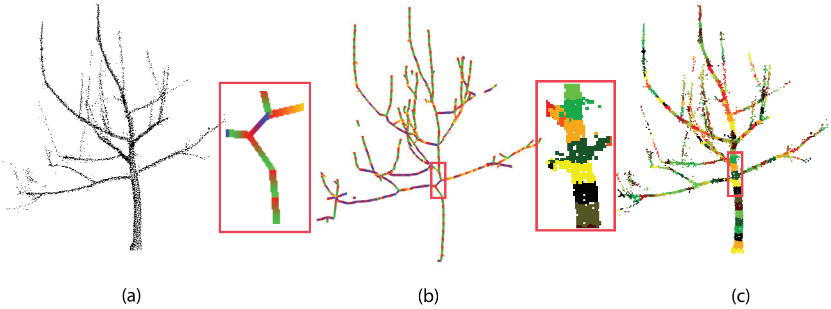
The resulting octree-graph should be interpreted as a bidirectional graph, as every edge gets two labels (Figure 4.1). Suppose, that two vertices $v_1$ and $v_2$, with Cartesian coordinates $(x_i, y_i, z_i)$ are connected. Let $x_1 < x_2$ and $y_1 = y_2$ and $z_1 = z_2$. Then the edge $e_{12}$ gets the labels $(1, 0, 0)$ and $(-1, 0, 0)$. Note that the sum of the labels belonging to one edge is the zero-vector (0,0,0) in 3D.

Depending on the choice of $f$, the resulting skeleton graph is augmented by vertices between the saddle points, minima and maxima of the surface. The minima and maxima of the surface are represented as vertices that have one incident edge in the skeleton graph and the saddle points generate the branching points having vertices with 3 or more incident edges. Note here that every vertex in the skeleton graph corresponds to a contour (compare Figure 4.4(b) and (c)) derived with respect to the local surface elongation. Figure 4.4(c) is comparable to the description in Section 2.3. Especially the branching object part in the zoom-in (yellow) has to be noticed, because it is very similar to the example given in Section 2.3. The non-branching object parts are given as many single contours along the branches.

## 4.3 Graph reduction

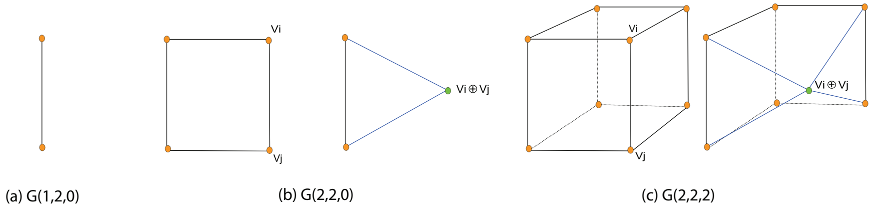
The extracted skeleton contains a known topological structure. This structure is the Reeb-graph, which was first introduced by [Reeb, 1946] and is strongly linked to Morse theory, [Milnor, 1963].

The input to the graph reduction is a labeled octree graph. Based on the introduced concept of octree-graphs, Section 4.2, and the underlying concept of *grid graph* (Def-



(a) (b) (c)

Figure 4.4: (a) Point cloud of a tree (b) Derived skeleton graph with color labeled direction labels and a magnification of the skeleton. All negatively labeled ends are colored red. The directions Up/Right/Front are colored with green/blue/yellow. (c) Each color segment corresponds to a subset of the point cloud. Each subset is associated to one contour and is corresponding to one vertex in the skeleton graph.



(a) G(1,2,0) (b) G(2,2,0) (c) G(2,2,2)

Figure 4.5: (a) a G(1,2,0) subgraph, belonging to the skeleton. (b) a G(2,2,0) subgraph and a possible pair to be merged derived from it. (c) a G(2,2,2) subgraph with an example of a possible pair to be merged. ⊕ denotes the merge of two neighboring vertices.

inition 4.2.3), graph reduction rules for the skeletonization algorithm are described. Graph reduction is done by merging suitable pairs of neighboring vertices, as specified below. The merging operation of two vertices will be denoted as ⊕.

The operations on the octree-graph are intuitively explained as the merge of two vertices incident to the same edge as shown in Figure 4.5 for the grid graphs present in an octree-graph. To quantify the reduction we introduce the *vertex dimension*, denoted as *vdim*.

**Definition 4.3.1.** *The number of distinct edge labels associated to a vertex* $v_i$ *is called the dimension* $vdim(v_i)$ *of a vertex.*

In Figure 4.1, the vertex in cell A has dimension 0, while the vertices in cell D and E have dimension 1. The vertices in cell B and C both have dimension 2. The vertex in cell C has two associated labels corresponding to two direction labels of opposite direction.

In 3D, the dimension of a vertex is at most 6. The convergence towards the skeleton and the preservation of the shape elongation is established by the notion of a local direction. This direction is defined per vertex as *vertex direction vdir*.

**Definition 4.3.2.** *The sum* $vdir(v_i)$ *over the distinct associated edge labels of a vertex* $v_i$ *is called the vertex direction.*

Each label in 3D is a 3D vector, which allows adding up the labels. In Figure 4.1 the vertex $v_1$ in cell B, with the associated labels $(0, -1, 0)$ and $(1, 0, 0)$ has vertex direction $vdir(v_1) = (1, -1, 0)$, which results from adding up the labels of the two edges. The vertex $v_2$ in cell C has $vdir(v_2) = (0, 0, 0)$.

The vertex direction encodes local surface elongation properties.

**Definition 4.3.3.** *Let* $x_1, x_2, x_3$ *be the three components of* $vdir(v) = (x_1, x_2, x_3)$. *A direction of* $x_i$ *is trivial if the Cartesian entries are zero for each associated edge label. A direction* $x_i$ *is dominant at* $v$*, iff* $x_i = 0$*, but not trivial. A direction corresponding to a non-zero value for* $x_i$ *is called a non-dominant direction.*

For example, in Figure 4.6a the dominant direction of vertex $v_k$ is marked green.

A set of definitions is now given for valid vertex merging. A special case applies to vertices of dimension 3. Consider the minimal configuration of G(2,2,0) subgraphs as given in Figure 4.6a with three vertices $v_i, v_j, v_k$, all three having vertex dimension 3. In Figure 4.6 it is visible that a merging operation removes one edge and its labels from the graph locally, and that all other labels are kept. In the framework given so far, it might happen that $v_j$ is merged with $v_k$, before $v_i$ is merged with $v_k$, resulting in the loss of a dominant direction. Figure 4.6b demonstrates such a leakage, where the dominant direction in $v_k$ is not represented anymore in the resulting merged vertex $v_i \oplus v_k$.

A solution to this problem is to take the *norm of a vertex* into account.

**Definition 4.3.4.** *Let* $vdir(v_i) = (x_1, x_2, x_3)$. *The norm of* $v_i$ *is*

$$norm(vdir(v_i)) = norm(x_1, x_2, x_3) = |x_1| + |x_2| + |x_3|$$

In case of vertex dimension 3, vertices with smaller norm value are reduced first. Taking the norm into account results in an unchanged dominant direction as can be seen in Figure 4.6c. Because $norm(v_k) = 1$ and $norm(v_j) = 1$ and $norm(v_i) = 3$, (Figure 4.6a), $v_k$ and $v_j$ are merged before $v_i$.

The local reduction of the graph will be driven by rules exploiting the underlying grid graph. Two special configurations are considered: so-called *E-Pair*s and *V-Pair*s. The two configurations are shown in Figure 4.7 and Figure 4.8. The definition of E-pairs and V-Pairs find their origin in the grid graph properties of the octree-graph. Because of the underlying octree organization the octree-graph is a collection of various connected grid graphs, as introduced in Definition 4.2.3. The primary goal of the graph reduction is to remove the overrepresented graph parts from the graph by merging vertices. These subgraphs are G(2,2,0) and G(2,2,2) grid graphs, forming loops (Figure 4.5).

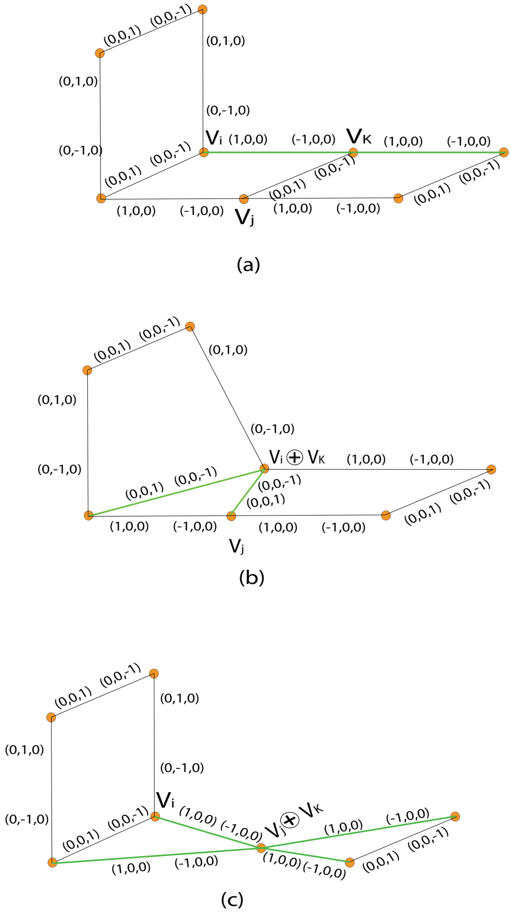


Figure 4.6: The dominant direction in $v_k$. (a) the minimal configuration. (b) a possible merge of $v_i$ and $v_k$ without taking the norm value into account resulting in a changed dominant direction (c) the merge of $v_j$ and $v_k$ preserving the dominant direction

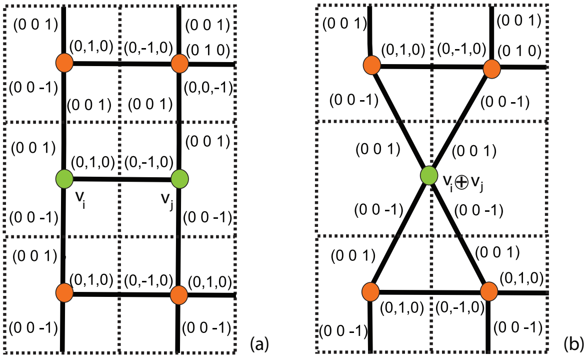


Figure 4.7: Circles indicate the vertices. The E-Pair and its merging are indicated in green. The dotted lines denote the cell sides and the labels are shown along the black edges. Note here that the position of $vdim(v_i \oplus v_j)$ is chosen arbitrarily. (a) an E-Pair configuration and (b) a merging result of the E-Pair in (a).

**Definition 4.3.5.** *Let $v_i$ and $v_j$ be two adjacent vertices with $vdim(v_i) \leq vdim(v_j)$. Then $v_i$ forms an E-Pair with $v_j$ if:*

1. $vdim(v_i \oplus v_j) \leq \max(vdim(v_i), vdim(v_j))$*;*
2. $vdir(v_i) \neq (0,0,0)$ *and $v_i$ and $v_j$ are connected in a non-dominant direction of $v_i$;*
3. *$v_i$ and $v_j$ are not a part of a $G(1,2,0)$ subgraph.*

In Figure 4.7(a) vertices $v_i$ and $v_j$ form an E-pair. The merging operation of $v_i$ and $v_j$ removes one edge and its labels locally from the graph. All other labels stay unchanged. The dimension $vdim(v_i \oplus v_j) = 2$, while $vdim(v_i) = 3$ and $vdim(v_j) = 3$. In Figure 4.7(b) vertices $v_i$ and $v_j$ are merged to $v_i \oplus v_j$.

**Definition 4.3.6.** *Two vertices $v_i$ and $v_j$ both incident to a vertex $v_c$ are called a V-Pair if,*

1. *the labels of edges $v_i v_c$ and $v_j v_c$ are identical;*
2. $vdim(v_i \oplus v_j) \leq \max(vdim(v_i), vdim(v_j))$.

The two vertices $v_i$ and $v_j$ shown in Figure 4.8(a) form a V-pair, because the edge labels $v_i v_c = (0,0,-1)$ and $v_j v_c = (0,0,-1)$ are identical and $vdim(v_i \oplus v_j) = 3$ is smaller than $max(vdim(v_i) = 3, vdim(v_j) = 4) = 4$. Figure 4.8(b) shows the resulting labels of $v_i \oplus v_j$. The operation $v_i \oplus v_j$ eliminated the two labels of the edge $v_i v_j$.

### 4.3.1 Topological Preservation and Metric Representation

In this paragraph we show that by systematically merging E-Pairs and V-Pairs an initial skeleton graph is obtained. This skeleton graph meets the requirements posed in Sec-

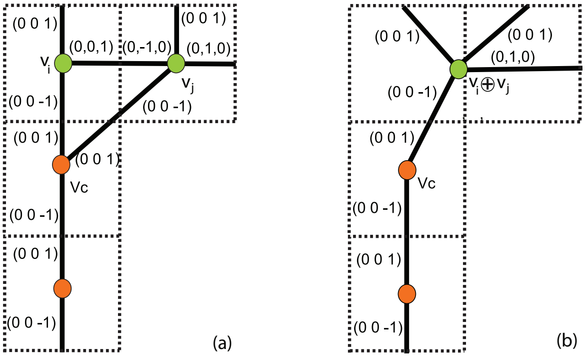


Figure 4.8: Circles indicate the vertices. The V-Pair and its merging are indicated in green. The dotted lines denote the cell sides and the labels are shown along the black edges. Note here that the position of $vdim(v_i \oplus v_j)$ is chosen arbitrarily. (a) Example of a V-Pair configuration and (b) a merging result of the V-Pair in (a).

tion 2.3. In order to simplify the following explanations we assume ideal conditions for the principal proof. It is assumed that first the point cloud is without noise and second, that it is sufficiently dense in the sense that it covers every directional change of the surface (see the concepts introduced in Chapter 2). The octree-graph extracted from the octree cells encodes both topology and surface information under these idealized conditions. We conclude, that the octree-graph is an alternative representation of the directional behavior of the sampled surface, embedded into 3D Euclidean space.

The octree-graph, consisting of its vertices and edges labeled by direction, is ideally to be retracted to a graph embedded into 3D-Euclidean space containing only loops representing the topological *genus* of the sampled object. At this stage we are ready to connect the resulting skeleton graph to the properties of a known skeleton graph, the so-called Reeb-graph.

Given a piecewise linear function $f$, the level set of a value $s \in \mathbb{R}$ is defined as the set of points with a function value equal to $s \in \mathbb{R}$. Recall that the octree-graph is embedded into 3D Euclidean space, and because of that every vertex has 3D-coordinates. The construction of a Reeb-graph is based on the analysis of the evolution of the connected components of the level sets generated by $f$. We call such a level set to be merged a contour. In the following we show, that the given retraction rules are an analysis of the evolution of the connected components of the level sets generated by $f$. Practically, a Reeb-graph connects the contours with respect to $f$. For a function value $s_i$ of $f$ where the number of contours increases compared to $s_{i-1}$, the Reeb-graph will split and indicate a saddle point of $f$. For values $s_i$ having no successor $s_{i+1}$ the Reeb-graph represents a maximum. A minimum is present if $s_i$ has no predecessor $s_{i-1}$.

In this paragraph we use a local elongation function $e(vdir(v))$ on the octree-graph as the piecewise linear function $f$. Here, $e(vdir(v))$ gives the local elongation of the object surface at a vertex $v_i$ by its vertex direction $vdir(v_i)$.

**Definition 4.3.7.** *Let* $v = (x, y, z)$ *be a vertex in* $\mathbb{R}^3$, *with* $x, y, z \in \mathbb{R}$. *Let* $vdir(v) = (d_1, d_2, d_3)$ *with* $d_i \in \{-1, 0, 1\}$, *where* $d_i = 0$ *indicates a dominant direction. The function to extract the contours is then defined as,*

$$e(vdir(v)) \quad = \begin{cases} x & \text{if } vdir(v) = (0, *, *); \\ y & \text{if } vdir(v) = (*, 0, *); \\ z & \text{if } vdir(v) = (*, *, 0). \end{cases} \qquad \text{with } * \in \{-1, 0, 1\}$$

Note that this definition is not unique in case there is more than one dominant direction. This, however, is not a problem. A different choice of dominant direction only corresponds to a different order of reduction of the octree-graph. Careful investigation of the elongation function $e(vdir(v_i))$ reveals that the height function is included as $e(vdir(v_i)) = z$, if the object is only elongated in the Cartesian z-direction.

**Proposition 4.3.8.** *Let OG be an octree graph derived from a sampled object. The merging operations on OG defined by E-Pairs and V-Pairs result in a skeleton containing the Reeb-graph defined by the piece wise linear function of Definition 4.3.7.*

We prove Proposition 4.3.8 by proving three Lemmas. First we prove (Lemma 4.3.9) that the local elongation is correctly represented and the dominant direction stays unchanged during merging operations of E-Pairs. Then we prove in Lemma 4.3.10, that E-Pairs always result in V-Pairs, which assures the convergence of the algorithm towards a skeletal line. At last we prove in Lemma 4.3.11 that merging V-Pairs does not change the dominant direction. Lemma 4.3.11 relies on the correct representation of the elongation in Lemma 4.3.9 and the derived convergence in Lemma 4.3.10.

Let OG be an octree-graph derived from an octree subdivision, as described in Section 4.2.2, formed of G(1,2,0), G(2,2,0) and G(2,2,2) subgraphs. These three underlying subgraphs are the minimal cases considered to describe locally a merge of two connected vertices. A G(1,2,0) is the trivial case belonging to the skeleton demanding no further processing (see Figure 4.5(a)). Every G(2,2,2) subgraph (Figure 4.5(c)) is the union of vertices and edges of six G(2,2,0) subgraphs (Figure 4.5(b)) inducing four valid E-Pair configuration. This coherence between E-Pairs and the G(2,2,0) grid graph allows us to prove Lemma 4.3.9 and Lemma 4.3.10 on a valid E-Pair configurations in a G(2,2,0) setting.

Recall, that every edge of *OG* is labeled with two labels, containing a positive and a negative component. In the following lemma we make use of the fact that two *directed graphs* can be derived from the octree-graph. The first directed graph is obtained by removing all edge labels from the octree-graph (Figure 4.9(a)) containing a positive entry (Figure 4.9(b)). The second directed graph is derived by removing all edge labels from the intermediate graph containing a negative entry (Figure 4.9(c)). The arrows in Figure 4.9(b) and Figure 4.9(c) show the direction of the edges. Note here that the directions are opposite to each other for the corresponding edges in Figure 4.9(b) and Figure 4.9(c).

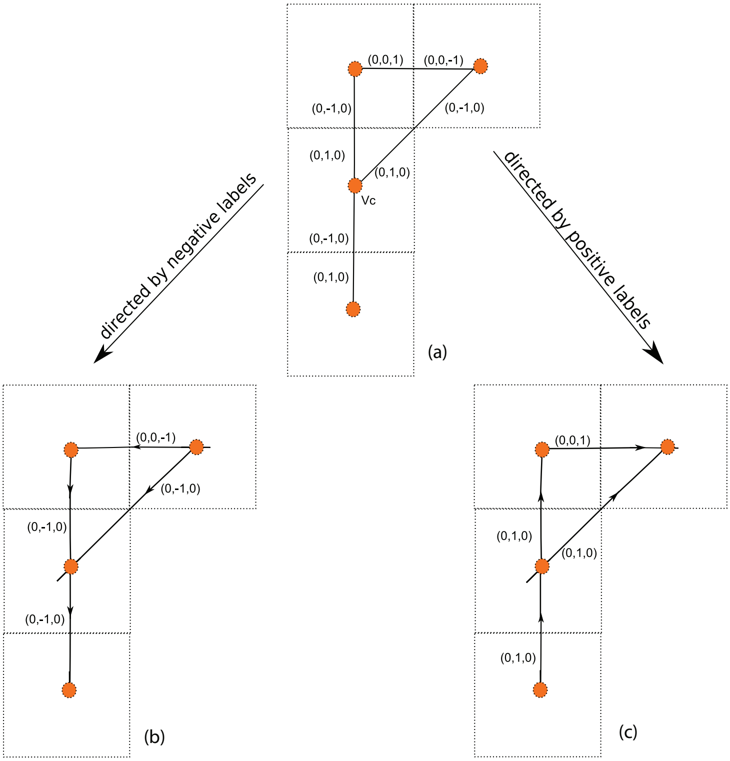


Figure 4.9: (a) a labeled octree graph is shown. In figures (b) and (c) the two derived graphs are shown. These derived graphs are directed. (b) contains only label elements ≤ 0 and (c) contains only label elements ≥ 0.

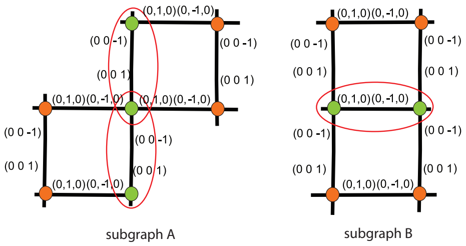


Figure 4.10: The two configurations considered in Lemma 4.3.9, with possible E-Pairs marked by an ellipse.

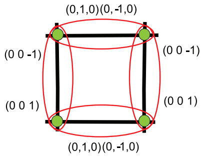


Figure 4.11: The configuration considered in Lemma 4.3.10, with possible E-Pairs marked by an ellipse.

**Lemma 4.3.9.** *Let E be an E-Pair consisting of two vertices $v_i$ and $v_j$. The merging operation $v_i \oplus v_j$ preserves the dominant directions of $v_i$.*

*Proof.* Let A and B be two connected G(2,2,0) subgraphs. Two cases have to be considered. First the case where A and B share exactly one vertex resulting in up to 2 valid E-Pair configuration, corresponding to exactly one dominant direction (Figure 4.10 subgraph A). The second case consists of two subgraphs sharing exactly one edge and 2 vertices. The second case results in exactly one possible E-Pair configuration (Figure 4.10 subgraph B). The E-Pair configuration of the subgraph contains only one non-dominant direction. According to Definition 4.3.5, merges only occur in non-dominant directions, preserving the dominant one. □

In the example given in Figure 4.10 subgraph A, two E-Pairs are visible, because the configuration is symmetric. The proof however is valid, since only one E-Pair is treated at a time, whose merge will let the other E-Pairs vanish. Note that an example of the elongation description by an E-Pair was already given in Figure 4.6(a).

The octree-graph is retracted by merging vertices forming a V-Pair. If no V-Pair is present in the graph to be retracted, a V-pair is created by an E-Pair representing local elongation of the sampled surface. Now that it is shown that the merge of an E-Pair preserves the elongation, it has to be shown that every E-Pair results in a V-Pair, for continuation of the retraction process.

**Lemma 4.3.10.** *The merging of an E-pair results in at least one V-Pair.*

*Proof.* A G(2,2,0) contains 4 valid E-Pair configurations, as illustrated in Figure 4.11. Consider one arbitrary E-Pair configuration of a G(2,2,0). Merging the two vertices involved in the E-Pair, reduces the squared structure to a triangle satisfying the definition of a V-Pair according to Definition 4.3.6. □

As every E-Pair results in a V-Pair, as shown in Figure 4.6(c) on an example, it is finally necessary to show that the merge of a V-Pair does not change the dominant direction in the graph.

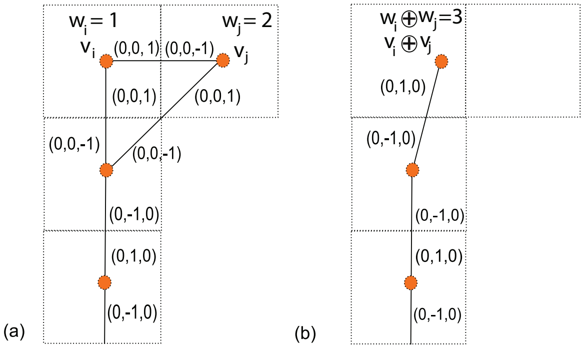


Figure 4.12: Example of weighted merging of vertices $v$ with $(x, y, z)$ coordinates: (a) shows a graph before merging is applied to $v_1$ and $v_2$ and (b) the graph with the new vertex $v_{new}$

**Lemma 4.3.11.** *Merging of V-Pairs does not change the local dominant direction.*

*Proof.* Consider the triangle structure of a V-Pair, e.g. the V-Pair formed by $v_c, v_i$ and $v_j$ and is identified by the two edges incident to $v_c$, both labeled with (0,0,1) in Figure 4.8(a). The triangle structure of a V-Pair contains at least one direction label exactly two times in a directed intermediate graph, and the label to be removed by the merging operation exactly one time. The merge removes exactly one dominant direction label, which preserves the difference between the amount of dominant and non-dominant labels. □

## 4.4 Graph embedding

The metric representation of a skeleton requires centeredness, which is achieved via a graph embedding strategy. Within this strategy, the octree-graph is embedded into the point cloud by averaging the points $\Sigma_i$ belonging to an octree-cell. The embedding introduced in [Pascucci et al., 2007] was adapted to point clouds, because their embedding can be updated during the computation of the SkelTre Skeleton. Every vertex in the octree-graph has an initial weight $w_i$ which is equal to the number of points belonging to the corresponding octree cell. During the merging process the weighted average of the 3D coordinates of two merged vertices $v_i$ and $v_j$ is taken to obtain the coordinates of the merged vertex $v_i \oplus v_j$. The weight $w_i \oplus w_j$ of $v_i \oplus v_j$ is then the sum of the previous weights (see Figure 4.12) that is $w_i \oplus w_j = w_i + w_j$. These weights are used to compute the coordinates of $v_i \oplus v_j$ into the point cloud.

The position of $v_i \oplus v_j$ is calculated as:

$$(v_i \oplus v_j) = \frac{w_i \cdot \ v_i + w_j \ \cdot v_j}{w_i + w_j} \tag{4.1}$$

In case of different local octree depth, the local octree depth is multiplied by the weight to obtain a centered skeleton as a result.

Embedding can be problematic if the boundary of $\Sigma_i$ is concave, because in that case the position of the centroid is not necessarily equal to the weighted average described above. In case of trees this occurs on vertices, where the skeleton branches. For such cases a post-processing step is required. The post-processing step treats the 3 or more connected vertices of the skeleton, by investigating the distance of a point subset to its bounding box. Consider the bounding box of some subset of points $\Sigma_i$. Let the vectors pointing from the center of the bounding box to the centers of each of the 6 sides of the bounding box be the 6 Cartesian directions. Each side corresponds to a Cartesian direction and because of that we select the value of the $\{x, y, z\}$-coordinate component corresponding to the side a point $p_i$ is closest to. The corrected vertex coordinate is computed by averaging the selected $\{x, y, z\}$ values for every coordinate component. Referring to Section 2.4, the embedding of the graph is a collection of ordered vertices placed at the centroids of point cloud subsets to obtain a useful metric representation. The case of a concave surface part crossing the octree cell is taken into account, because a misplaced initial centroid in the octree-graph would not result in a properly placed centroid during merging, as explained above, which would make later analysis difficult. The motivation for a properly placed centroid was already given in Section 2.4.

## 4.5 Summary

In this chapter the rules driving the SkelTre skeletonization method have been presented. The rules are motivated by the descriptions in Chapter 2, which concluded that a skeleton with all four postulated requirements is derivable if we can extract local elongation directions and use their changes to derive a graph whose vertices are located at the center of gravity of suitable subsets of the point cloud. Here we related the extracted graph to the surface based on a locally chosen function, representing Cartesian elongation directions. The presented method reduces an initial graph by a set of rules acting on the Cartesian elongation directions to the SkelTre skeleton. The topological preservation was shown by relating the extraction process to a well known topological structure, the Reeb-graph. The output of the algorithm is a graph, which is by definition thin, whose branching corresponds to the branching of the object with respect to the introduced elongation direction function and is connected based on the adjacent octree cells. Because of that the resulting graph satisfies the attributes of topological preservation introduced in Section 2.3. Furthermore the usage of a robustness criterion was introduced to improve the handling of imperfect data. This robustness criterion improves connectedness also under the presence of data gaps caused by measurement errors and occlusion effects and removes edges from the octree graph which are present because of outliers. The robustness criterion also allows the algorithm to use error models as desired by the skeleton requirements in Section 2.1.

The use of an octree makes SkelTre capable to operate hierarchically as described in the skeleton requirements for computational efficiency in Chapter 2. The potential of the method to skeletonize point clouds representing a much larger class of objects exists, because no restrictions to tree-like structures are made in the skeletonization rules. Furthermore, an adoption of an existing graph embedding into point clouds was introduced to achieve correct metric representation by embedding the skeleton-graph vertices as local centroids into the point cloud. The metric representation is derived from a known embedding strategy adapted to point clouds. The next chapter will empirically analyze the algorithm performance and discuss the implementation of the SkelTre-algorithm.

# 5

# Analysis of the SkelTre Algorithm

In Chapter 2 we introduced two descriptive requirements (*topological preservation* and *metric representation*) of a skeleton and two requirements of a skeletonization algorithm (*computational efficiency* and *reliability*) operating on a point cloud. Until here we have shown that the topological preservation requirement is met in the design of the SkelTre algorithm (Chapter 4). This chapter demonstrates the *implementation* of the SkelTre algorithm as pseudo code. Beside that, the *computational complexity* of the SkelTre-algorithm is discussed to verify the efficiency requirement introduced in Chapter 2 along with the algorithms behavior per vertex dimension as function of the number of vertices in practice. The embedding of the extracted skeleton graph, as introduced in Chapter 4, depends mainly on the data and is validated empirically by investigating the attributes of skeleton requirements introduced in Chapter 2. This empirical validation of the embedding gives a detailed view on the desired centeredness and the behavior under rotations of the resulting skeleton, as defined in Chapter 2. Additionally, the algorithm is evaluated on a test tree containing the typical problems of real data (compare Chapter 2) against the CAMPINO method [Bucksch and Lindenbergh, 2008] introduced in Chapter 3. The results of this chapter are an extended collection of the results in [Bucksch et al., 2009a] and [Bucksch et al., 2010].

In Section 5.1 the *computational efficiency* of the *SkelTre* algorithm, the *SkelTre* implementation and the SkelTre behavior is discussed. Section 5.2 validates the centeredness on real data examples from different scanner sources with different *error characteristics*. Section 5.3 summarizes the highlights of the chapter.

## 5.1 Algorithm Efficency

This section discusses an implementation of the given graph reduction (compare Procedure 1 and Procedure 2). The computational efficiency of the SkelTre skeletonization is evaluated in Paragraph 5.1.1. Insight into the algorithm behavior is given by investigating the graph reduction per dimension in Paragraph 5.1.2, for the vertex dimensions (Definition 4.3.1) ranging from $n = 5$ to $n = 2$. The following three steps have to be implemented in a sequence, and are illustrated on a simple example in Figure 5.1.

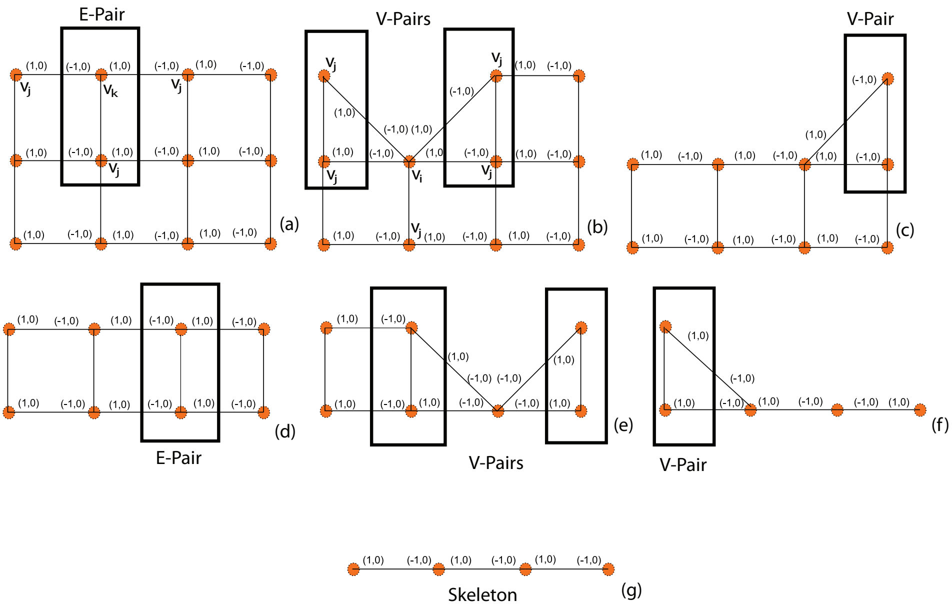


Figure 5.1: (a) An input grid graph with one possible E-Pair marked. The vertices $v_j$ denote the adjacent neighbors of $v_k$ (b) The merged E-Pair results in two V-Pairs. The vertices $v_j$ denote the adjacent neighbors of $v_i$ (c) The two merged V-Pairs of (b) result in one more V-Pair to be merged. (d) Another E-Pair selected for merging. (e) The merge of the E-Pair selected in (d) results in two V-Pairs. (f) The merging operation on the two V-Pairs in (e) results in one more V-Pair. (g) After merging the V-Pair formed in (f) the final skeleton graph is obtained.

1. Initialize a vertex list *dimList* containing all vertices of dimension $n$. For each vertex $v_i$ in *dimList* test if new V-Pairs can be formed with its adjacent neighbors $v_j$, until either a V-Pair is found, or no direct neighbors to test are left. Such a initial check is necessary if a starting configuration, as shown in Figure 5.1 (b), is the input graph. All V-Pairs found are stored in *PairList*.

2. If during the loop through the vertex list *dimList* no V-Pair was found, go through the vertex list *dimList* again. Whenever an E-Pair is found (Figure 5.1 (a)) for vertex $v_k$, say, the E-Pair is merged to a new vertex $v_i$ (Figure 5.1 (b)). This results according to Lemma 4.3.10 in one or more V-Pairs with the direct neighbors of $v_i$ (Figure 5.1 (b)). Repeat this procedure until all entries of *dimList* are processed (Figure 5.1 (c)-(g)).

3. All vertex pairs in the *PairList* are merged. Directly after merging it is tested whether the merging resulted in the creation of new V-Pairs (Figure 5.1 (b),(c),(e) and (f)). If so, these V-Pairs are added at the end of *PairList*.

Remember that the merging of two vertices $v_i$ and $v_j$ along a common edge is denoted by $v_i \oplus v_j$. The resulting merged vertex $v_{new} = v_i \oplus v_j$ inherits all incident edges from its ancestors $v_i$ and $v_j$. If $v_i$ and $v_j$ were both incident to a common vertex $v_c$, then the two edges $v_i v_c$ and $v_j v_c$ are collapsed to a common edge $(v_i \oplus v_j)v_c$. Under ideal conditions these edges represent the connection between two connected subsets of the sampled surface $\Sigma$. For this reason, the operation $v_i \oplus v_j$ is only performed on vertices representing two neighboring subsets of $\Sigma$ with the same direction characteristic, as indicated by the identical edge labels of $v_i v_c$ and $v_j v_c$.

### 5.1.1 Computational Complexity

In practice, the computation of the skeleton operates on a far smaller number of vertices than the number of points in the point cloud. We explain here , that the graph-reduction of the SkelTre algorithm is linear in time. A pseudo code to implement the algorithm is shown in Procedure 1 and Procedure 2.

Let $v_i$ be a vertex of the set of vertices $V$ of the processed graph. The dimension of $v_i$ is denoted as $vdim(v_i)$. Let $v_j$ denote an adjacent vertex of $v_i$ and $c_1 \leq 6$ a constant corresponding to the maximal number of vertex dimensions. The procedure *computeSkeleton* contains an outer for-loop, which is bounded by $c_1$ and therefore *O(1)*. As can be noticed in Procedure 1, *dimList* is always initialized with *O(V)*. Note that $V$ is decreasing after every dimension. The inner while-loop is operating on a subset of $V$ with at most $\frac{V}{2}$ operations, which results in $O(\frac{V}{2})$ as an upper bound.

The procedure *createVPair()* selects the first unprocessed entry $v_i$ in *dimList*, that fulfills Definition 4.3.5, and loops through all elements of *dimList*. We show the influence of this condition on the inner while-loop for two extreme cases:

```
Input: An Octree graph
Output: SkelTre Skeleton
dimList[]; contains the vertices of the processed dimension
PairList[]; contains found EPairs and VPairs to be merged
for dim=5 to 2 do
    $dimList := \{v_i | vdim(v_i) \geq dim, i = 0...n\}$;
    forall $v_i \in dimList$ do
        if $IsVPair(v_k, v_i)$ for some incident vertex $v_k$ of $v_i$ then
            add $(v_i, v_j)$ to the end of PairList;
        end
    end
    while $PairList \neq \emptyset$ or $createVPair()$ return true do
        $\{v_i, v_j\} = first\ unprocessed\ entry\ in\ PairList$;
        $v_{new} = v_i \oplus v_j$;
        if $(IsVPair(v_{new}, v_k)$ for some incident vertex $v_k$ of $v_{new}$ then
            add $(v_{new}, v_k)$ to the end of PairList;
        end
        if $vdim(v_{new}) \geq max(vdim(v_i, v_j))$ then
            add $v_{new}$ to the end of dimList;
        end
        remove $\{v_i, v_j\}$ from PairList;
    end
end
```

**Procedure 1** `computeSkeleton`

```
Input: a unprocessed vertex $v_i$ from dimList
Output: true or false
if $vdir(v_i) \neq (0, 0, 0)$ then
    foreach $v_j$ adjacent to $v_i$ do
        if $IsEPair(v_i, v_j)$ for $\{v_i, v_j\}$ then
            add $(v_i, v_j)$ to the end of PairList;
            return true;
        end
    end
end
return false
```

**Procedure 2** `createVPair`

- **Case 1:** The input graph is already a skeleton, e.g. a combination of G(2,1,0) subgraphs, all connected on only 2 vertices. This combination will lead to no merge at all; because of that the whole inner while-loop of *computeSkeleton()* stays *O(V)* by checking one time if there is any valid E-Pair configuration (Definition 4.3.5).

- **Case 2:** All $vdim(v_i)$ within $V$ are equal, e.g. a graph formed by G(2,1,0) subgraphs, which all are connected on three vertices. This will lead to exactly one check of an E-Pair (Definition 4.3.5) in the given example, and $c_2$ calls in general. $c_2$ is bounded by the minimal number of aligned G(2,2,0) subgraphs in one of the principal Cartesian directions.

Now that it is shown that the influence of *createVPair()* on the inner while-loop is $O(c_2)$ or $O(V)$, the algorithms overall complexity can be calculated as follows from the upper bound; $O(1) \cdot (O(V) + O(\frac{V}{2}) + O(V)) = O(V)$. Note here, that the special case of dimension 3 vertices is simply handled with a *dimList* for every possible norm value. Because of that vertices of dimension 3 have no influence on the complexity analysis given here.

### 5.1.2 Algorithm behavior

Condition 3 in Definition 4.3.5 ensures convergence towards the skeleton. Vertex dimension 6 results in vertex direction (0,0,0) in all cases. This characteristic of dimension 6 vertices allows the successive reduction of the octree-graph from vertices of dimension 5 to 2. This guarantees that first the $G(2,2,2)$ graph parts are reduced, before $G(2,2,0)$ subgraph regions are processed. Another example to depict the algorithm is to evaluate the number of vertices per processed dimesion. Boundary vertices of a $G(2,2,2)$, which have dimension 4, are processed before the 'corner'-vertices of a $G(2,2,2)$ area, and the boundary of a $G(2,2,0)$ region is processed before its 'corners' of $vdim(v) = 2$.

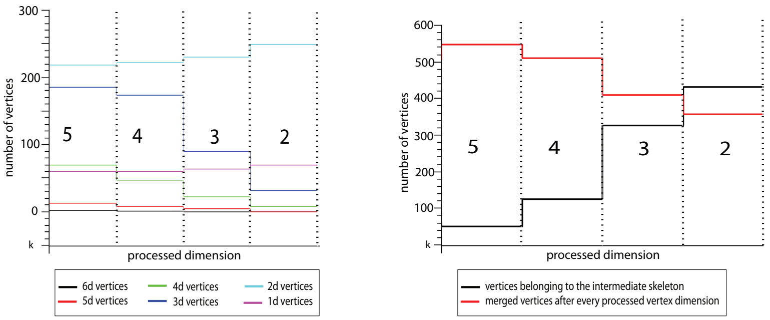


Figure 5.2: (Left) amount of vertices per dimension after every processed vertex dimension. (Right) The red graph shows the vertices belonging to the intermediate skeleton and the black graph the number of merged vertices after every processed vertex dimension.

In Figure 5.2 the *convergence* of the SkelTre algorithm is illustrated. The values correspond to the skeleton in Figure 4.4(b). Figure 5.2 (left) depicts the algorithmic behavior of SkelTre per vertex dimension. It is observable, that the amount of *vdim*(1) and *vdim*(2) vertices is increasing, as they mainly form the targeted skeleton. Meanwhile, *vdim*(5) and *vdim*(6) vertices are vanishing as expected, also a rapid decrease of *vdim*(3) and *vdim*(4) vertices is recognized. This behavior of the algorithm is coherent with the behavior described in theory.

The red curve in Figure 5.2(right) shows that the number of graph vertices is decreasing after every processed vertex dimension. The black curve shows the overall number of merging operations after every processed vertex dimension, when the smallest change occurs between dimension 3 and 2 in this case. This approximate constant behavior is in this particular example explainable by the influence of the procedure *createVPair()* (compare Paragraph 5.1.1 Case 1), because this intermediate case is already close to the desired skeleton.

## 5.2 Results and Practical Validation

This section on the evaluation of the extracted skeletons considers several examples and validation indicators. The section is divided into three parts, trees, non-tree objects and the comparison with an algorithm from the same algorithm class, [Bucksch and Lindenbergh, 2008]. The point clouds evaluated in this section are scanned with a variety of scanners and have different density, sampling and noise characteristics. Limitations are pointed out to explain the algorithm behavior. In all cases the skeleton edges are colored by their resulting direction labels. Yellow denotes a (1,0,0) label, blue a (0,1,0) label and green a (0,0,1) label. In all labeling cases the corresponding negative label is indicated in red.

No *post processing* is applied to the skeletons computed with the SkelTre algorithm. *Centeredness* is analyzed by considering the (average) Euclidean distance of every point of the point cloud to the skeleton. First, results on botanical trees are presented and secondly results on non-tree-like objects are shown.

### 5.2.1 Trees

This paragraph shows the problems and benefits related to *terrestrial laser scan data*. Figure 5.3 shows the results of the SkelTre algorithm on point clouds representing three different trees scanned with three different terrestrial laser scanners: a Simple Tree, an Apple tree and a Tulip tree. For the Simple tree and the Apple tree, the maximum of the color scale indicates distances to the skeleton of 10cm. For the Tulip tree, the maximum of the color scale indicates distances to the skeleton of 100cm.

The Simple Tree was scanned with a Leica Scan Station and was previously used in [Gorte and Pfeifer, 2004]. This tree of 4.07m height is sampled by 49669 points. The skeleton graph resulting from the SkelTre algorithm is fully connected and all 18 branches were detected, even under bad sampling conditions (upper red box in Fig-

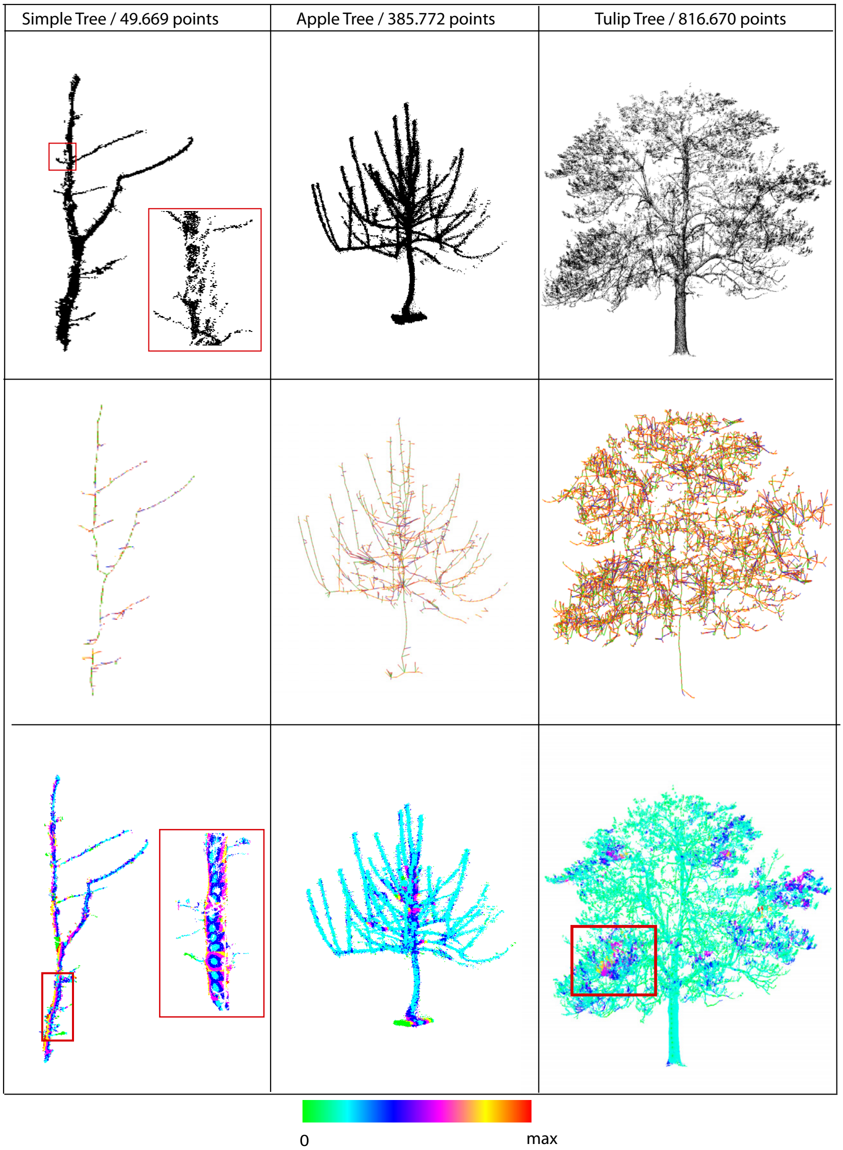

Simple Tree / 49.669 points
Apple Tree / 385.772 points
Tulip Tree / 816.670 points
0
max

Figure 5.3: First row shows the raw point cloud. The second row shows the skeleton colored by direction label. All negative ends are labeled red. The directions Up/Left/Front are colored by green/blue/yellow. The third row shows the distances to the skeleton according to the color scheme given on the bottom of the figure.

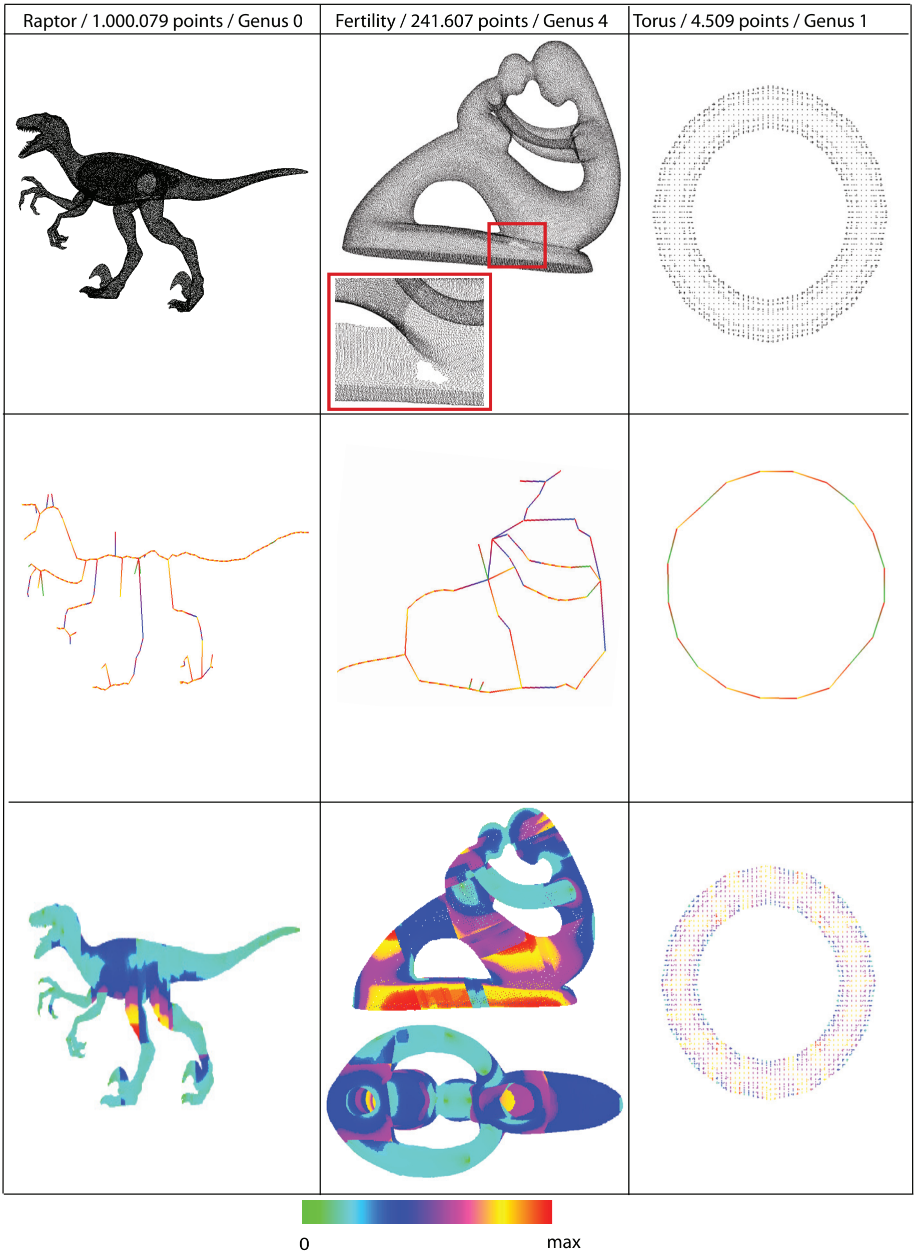

Raptor / 1.000.079 points / Genus 0
Fertility / 241.607 points / Genus 4
Torus / 4.509 points / Genus 1
0
max

Figure 5.4: Three test objects. First row shows the raw point cloud. The second row shows the skeleton colored by direction label. All negative ends are labeled red. The directions Up/Left/Front are colored by green/blue/yellow The third row shows the distances to the skeleton according to the color scheme given on the bottom of the figure.

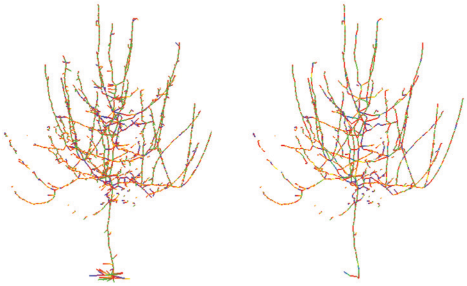


Figure 5.5: The extracted skeleton (Left) and the filtered skeleton (Right) of an apple tree

ure 5.3). The distances to the skeleton show a symmetric pattern in the red zoom box. The Simple Tree is represented by a single scan from one viewpoint. On this single scan of the Simple Tree in Figure 5.3 the skeleton is attracted to one side (light blue color), because the back of the tree was not scanned.

The Apple Tree was scanned with the Zoller and Fröhlich scanner Imager 5006 in high resolution (385.772 points, Stefan Fleck, University of Göttingen). The Apple tree is 1.99m high and its largest extension in the crown is 1.62m. The derived skeleton of the Apple tree contains only one erroneous loop due to unresolvable noise problems in the dense crown containing 136 detected branches. The distances to the skeleton get larger in the inner crown, because the crown contains a huge amount of noise. Spurious branches on the boundary indicate the noise on the object boundary. Simple removal of vertices with one incident edge adjacent to a vertex with three or more incident edges solves this problem in our experience, as shown in the example in Figure 5.5.

The Tulip Tree was scanned with a Calidus scanner. It is 11.75m high and 14.47m wide at the largest extension of the crown. This massive tree was scanned by Forstliche Versuchs- und Forschungsanstalt Freiburg near Karlsruhe, Germany. It is sampled by 816.670 points. The tree was scanned during summer time and contains leaves. Even under the presence of leaves the main skeleton was derived. The marked region indicates the difficulty on this particular tree. Noise, e.g. because of moving leaves during the scan procedure, makes a meaningful extraction of the skeleton impossible. The locally non-optimal extracted skeleton is visible as distances to the skeleton in the order of 5-8cm on fine branches. Such fine branches should be at a distance around 0.5-1cm to the skeleton to meet the centeredness attribute of the metric representation requirement in Chapter 2. This indicates that the chosen resolution of the octree is insufficient to resolve the structure. Still, the major branches and the trunk are extracted correctly and only locally the skeleton is affected by insufficient extraction.

### 5.2.2 Non-tree objects

In this paragraph the performance of the SkelTre algorithm on point clouds of non-tree objects is illustrated with three examples (Figure 5.4). Note here that distances are not given with a unit, because the unit of the data sets is not known.

The Raptor model with lots of ripples on its skin is taken from the aim@shape repository, [Aim@Shape, 2008]. It consists of 1 000 079 points. No information about the measurements to obtain the point cloud are provided by this web-source. The resulting skeleton graph is fully connected. The skeleton of the left foot shows an irregularity in the distances to the skeleton. This irregularity is explained by insufficient coverage of directional changes of the surface by the initial graph. Because of insufficient coverage the embedding is not completely centered.

The Fertility model is a genus 4 stone sculpture and taken from the aim@shape repository as well. It consists of 241 607 points. It was obtained with a Roland LPX-w50 laser range scanner. The marked region shows a huge hole in the point cloud of the statue, which prevented the use of fine resolutions. This results in insufficient skeleton resolution on the left and in the head region of the statue, because not all directional changes of the statue could be modeled by the octree graph. The genus is still represented correctly in the skeleton, and the extrema are modeled correctly as far the octree-graph covered them (e.g. on the baby's stomach).

The Torus model is sampled by 4 509 points from a polygonal approximation of a torus created with the 3D modeling package Lightwave 7.0. The sampling is rough to show the influence of strong undersampling.

### 5.2.3 Comparison with CAMPINO

The comparison with another algorithm of the same algorithm class, as introduced in Chapter 3, is given in this paragraph. Here the comparison is done between the SkelTre Skeleton, [Bucksch et al., 2010] and the CAMPINO skeleton, [Bucksch and Lindenbergh, 2008] on an imperfect point cloud of the example tree already used in Chapter 3. The example tree is subject to noise, undersampling, occlusion effects and varying point density in the indicated areas in Figure 5.6(a) and contains 7183 points. To give a fair comparison between the two graph reductions the same octree was used as input with an minimum cell size of 0.07m.

Figure 5.6(b) shows the extracted SkelTre skeleton and Figure 5.6(c) shows the extracted CAMPINO skeleton. It is already visible here, that the small height difference of the two lowest branches on the trunk is reconstructed by the SkelTre method, whereas the CAMPINO method was not able to successfully model this height difference. A geometric extraction using the height function would result in a non-centered skeleton, because some branches are almost horizontal. The distances of the points to the SkelTre skeleton never exceeded 4cm on the stem, while the CAMPINO methods shows distances above 4cm. In Figure 5.6(a) an important characteristic on scanned botanical trees is visible. The points in the strongly undersampled region I are in line-like order, which makes it hard to define a suitable inside and outside on the object, as required by some of the algorithms discussed in Chapter 3.

The improvement on the centeredness is quantified in Figure 5.6(d) by calculating the *histogram of point distances to the skeleton* binned in 1mm bins. Red represents the CAMPINO method and green the SkelTre method. The histograms are superimposed

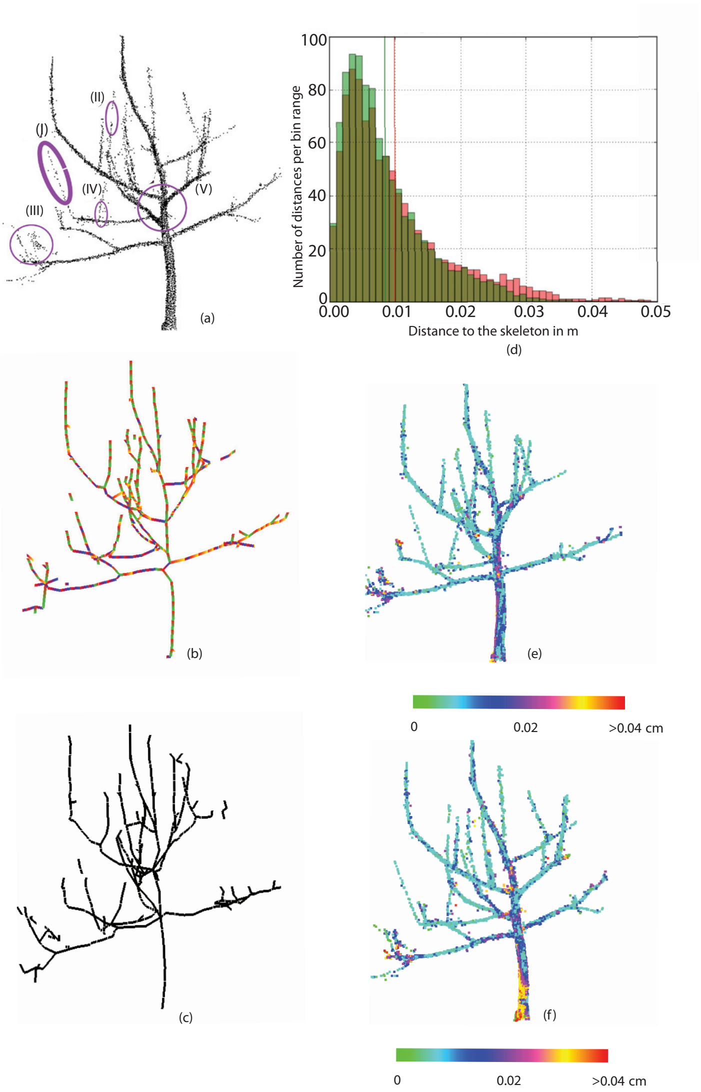


Figure 5.6: **(a)** The example tree marked with (I) strongly undersampled region. (II) data gap because of occlusion. (III) random noise because of combined effects. (IV) combined occlusion and undersampling (V) varying point density **(b)** the extracted SkelTre Skeleton with edges colored by their label. **(c)** a comparable result with [Bucksch and Lindenbergh, 2008]. **(d)** Comparison of the point distances to the skeleton between CAMPINO (red) and SkelTre(green). The vertical lines show the mean of the distances. **(e)** Centeredness as distances to the skeleton with CAMPINO **(f)** Improved centeredness with the SkelTre Algorithm.

for easier comparison. The improved centeredness is visible in both: The shift of the mean distance is shown as vertical lines. Using the CAMPINO method a mean distance to the skeleton of 0.01cm was achieved. The SkelTre skeleton improved the mean distance by 20% to 0.008cm. The lower achieved distances to the skeleton are an improvement on the centeredness attribute of the metric representation requirement in Chapter 2. Therefore, the improvement of 0.002mm is significant. The histogram shows more points at larger distance for the CAMPINO method and more points at smaller distances for the SkelTre method. The improvement on the centeredness is significant, because the diameter of the outer branches is below 5mm. Beside that it has to be noticed that also the calculation time has been significantly improved. While the CAMPINO method took approximately 10 minutes to extract the skeleton, SkelTre only took 34 seconds.

### 5.2.4 Behavior under rotations

In the previous chapter it was claimed, that approximate rotational invariance is achieved by investigating how the point cloud is passing through the octree cell sides. In the past, methods associated with a Reeb-graph extraction have been criticized for their rotational dependence in some publications, e.g. [Cornea and Min, 2007]. Here it is investigated how the SkelTre-skeleton extraction behaves under rotations. Figure 5.7 shows the extracted skeletons of the example tree (Figure 5.6) under rotations of 10, 25 and 40 degree around each Cartesian axis. The minimum cell size was fixed at 7cm. The skeleton is colored by the different direction labels as in the figures before. The skeleton extracted from the rotated point cloud, is rotated back to the original point cloud for comparison with the skeleton extracted from the unrotated point cloud. The black points on the skeletons in Figure 5.7 are the vertices of the skeleton extracted from the original unrotated point cloud. From Figure 5.7, it is qualitatively visible that there is high correspondence between the skeletons derived after rotating the point cloud first, because the black points representing the skeleton vertices extracted from the unrotated point cloud are on or close to the skeleton extracted from the rotated point cloud.

In Table 5.2.4 I list the results of the median Hausdorff distance as a quantitative validation, as specified below. Let $X, Y$ be two sets of vertices. The two sets are at median Hausdorff distance $d_H$ iff $d_H$ is the smallest number such that the vertices in $X$ are in median distance $d_H$ to all vertices in $Y$. $d_H$ is therefore given as

$$d_H(X,Y) = median\{\sup_{x\in X}\inf_{y\in Y} d(x,y), \sup_{y\in Y}\inf_{x\in X} d(x,y)\}, \tag{5.1}$$

where $d(x,y)$ is the Euclidean distance between two vertices $x$ and $y$ from the two sets. This measure evaluates the vertex to vertex distances between the two extracted graphs. The above comparisons have been chosen, because an intuitive comparison of the Euclidean median distance of the vertices of the rotated skeletons to the edges of the unrotated skeleton results in 0.0 cm in all cases given in Figure 5.7. Both distance evaluations indicate high robustness to rotations. The vertex to vertex correspondence

| Rotation around axis | Median Hausdorff distance in cm |
|---|---|
| 10°x | 0.02 |
| 25°x | 0.01 |
| 40°x | 0.01 |
| 10°y | 0.04 |
| 25°y | 0.01 |
| 40°y | 0.02 |
| 10°z | 0.01 |
| 25°z | 0.01 |
| 40°z | 0.04 |

Table 5.1: SkelTre under rotation: The Table shows the median Hausdorff distance.

shows variations up to 4cm which is approximately half of the minimum octree cell size.

### 5.2.5 Running time

The calculation time for the six examples given in Figure 5.3 and Figure 5.4 is given in Table 5.2. The table shows the size of the point cloud as number of points and its data range and the time needed to construct the octree under a fixed octree cell size. Furthermore, the number of octree graph vertices and the time needed to reduce the graph with the SkelTre algorithm are given. The calculation times given refer to a Intel Dual-Core processor 6700 running at 2.66GHz having 3.5GB memory. The operating system used was Windows XP with Service Pack 3 installed.

Table 5.2 shows the time required to perform the octree construction and the graph reduction. The *running times* are given for the algorithm as described in Section 5.1. The implementation uses the vector container of the Standard Template Library, [Musser and Saini, 1997] to store the adjacency list of the graph and all other intermediate lists, causing performance loss above 1000 graph vertices. The major reason why the skeletonization process slows down in our current implementation is that the used data structures make use of the page file on the hard disk at a certain amount of vertices. To prevent calculation losses because of the automatic shutdowns forced by the update procedures of the TU Delft ICT department, the algorithm writes an intermediate result every 500 merges of vertices to the hard disk. This hard disk access is an additional slow down, but the intermediate state of the skeletonization can be recovered at any time. The algorithm was tested on point clouds up to 4 Million points. However, its calculation depends more on the complexity of the object than on the amount of points itself. Once the octree-graph is extracted, the actual point cloud is not needed anymore.

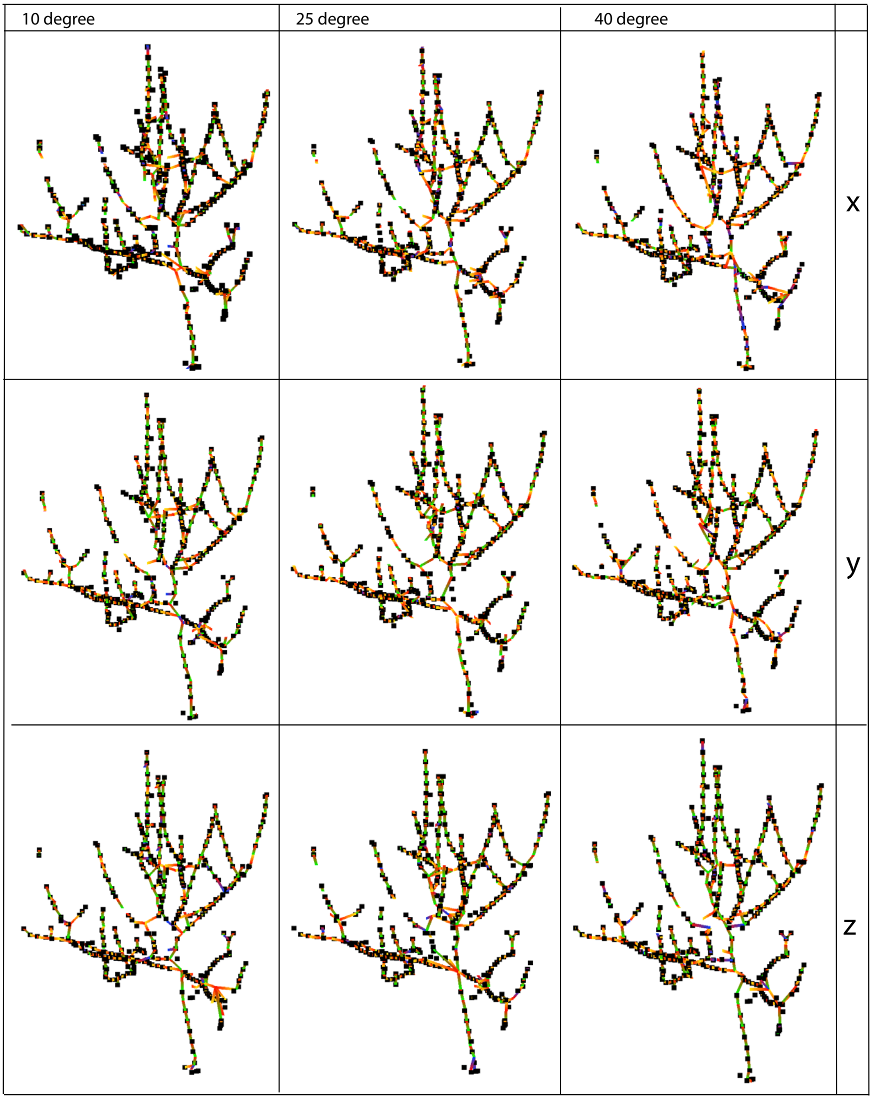


Figure 5.7: The shown skeletons are extracted with a minimum cell size of 7cm. The input data of the example tree was rotated by 10, 25 and 40 degree for each axis. The black points indicate the vertices of the skeleton extracted from the unrotated point cloud of the example tree. The skeleton colored by the different direction labels. All negative labeled ends of an edge are red. The directions Up/Left/Front are colored by green/blue/yellow.

| Model | Nr. of points | $T_{OG}$ in s | $T_{GR}$ in min | $OCV$ | $OCS$ | Data range<br>x<br>y<br>z |
|---|---|---|---|---|---|---|
| Simple Tree | 49 669 | 0.45 | 1.1 | 667 | 0.05 | [225.0-229.1]<br>[393.2-397.3]<br>[102.2-106.3] |
| Apple Tree | 385 772 | 2.7 | 192.0 | 5064 | 0.05 | [225.0-229.1]<br>[393.2-397.3]<br>[102.2-106.3] |
| Tulip Tree | 816 670 | 1.8 | 275.6 | 5084 | 0.5 | [225.0-229.1]<br>[393.2-397.3]<br>[102.2-106.3] |
| Raptor | 1 000 079 | 2.3 | 10.5 | 2404 | 0.04 | [225.0-229.1]<br>[393.2-397.3]<br>[102.2-106.3] |
| Fertility | 241 607 | 0.5 | 9.9 | 2132 | 10.0 | [-75.7-123.9]<br>[-76.9-122.8]<br>[-36.9-162.8] |
| Torus | 4 509 | 0.1 | 0.01 | 92 | 0.5 | [-1.2-1.2]<br>[-0.2-2.2]<br>[-1.2-1.2] |

Table 5.2: Running time for the example objects in Figure 5.3 and Figure 5.4. $T_{OG}$ denotes the time needed to construct the octree and $T_{GR}$ the time needed to reduce the octree-graph. $OGV$ denotes the number of octree-graph vertices and $OCS$ abbreviates octree cell size.

## 5.3 Summary

The chapter gave an evaluation of the algorithm in terms of computational efficiency as a complexity analysis and convergence graphs. It was shown that the graph reduction is linear in time, which underlines the efficiency of the method that is designed to skeletonize real, large point clouds of botanical trees and a wider range of object classes (compare algorithmic requirements given in Section 2.1). It was already shown in Chapter 4, that the algorithm is operating locally, which overcomes the spatial complexity problem stated in Section 2.6. The correct metric representation was analyzed by considering the influence of the embedding on the centeredness. Topological correctness, as proven before in Chapter 4 with respect elongation function, was described as the number of loops in the resulting graph of a tree and the resulting branches. The resulting number of loops are a consequence of the imperfect data. For non-tree object the genus was the classifying argument. Robustness has been shown on example point clouds containing the typical difficulties noise, undersampling and varying point density. Robustness as a prerequisite of the algorithmic requirements was shown on several selected examples and on the example tree containing the challenges of laser data previously used in Chapter 3. The transformation invariance was discussed by investigating the configuration of skeleton graph vertices on a rotated tree and the distance between an initial tree and the rotated tree, which demonstrates the desired transformation invariance of the metric representation (compare Section 2.2).

# 6

# Extraction of branch length and diameter of trees

In this chapter two applications of the SkelTre-skeleton algorithm are demonstrated. In Section 6.1 the extraction of the *branch parameters* length and diameter is done analyzing terrestrial laser scans of orchard trees. These *orchard trees* are optimized by branch pruning, which makes it impossible to use species dependent *allometric relationships* to estimate the tree structure. Hence, a measurement method is needed relying on surface information rather than prior knowledge to analyze orchard tree canopies. Here the frequency distributions of extracted branch lengths and diameters are analyzed.

The second application introduced is the extraction of stem diameters at breast height (1.30m) from strongly undersampled airborne laser data in Section 6.2. Until now, stem diameter estimation on airborne data was carried out by using allometric relations between the canopy size and the stem diameter of a **known** species. This application is potentially promising, as densities of airborne data are expected to increase even further in the coming years, which would allow the future use of skeletons on airborne data. In fact this case study reveals the need for improved *tree-delineation* methods, as the major problem is found in the lack of reliable tree-delineation methods.

Both sections start by describing the application fields where tree parameter extraction is expected to play a vital role. The first application is quantitative ecology and the second one is water flow management. Both described applications are unpredecented in their methodology and have to be reflected as a step towards automatic allometry free measurement on trees. Allometry free measurements on trees are predicted in literature e.g. [Strahler et al., 2008]. In order to go towards this ambitious goal both sections use the newly developed method HARPER - *H*istogram *A*nalysis & *R*etraction of *P*oints *E*stimated *R*adii - as a method to extract the diameter on the basis of a SkelTre skeleton.

## 6.1 Extracting branch length and diameter from terrestrial laser scanning point clouds of orchard trees

### 6.1.1 Introduction

Tree branches are the result of ramification and branch elongation processes that occur, outside the tropics, in an annual cycle. The pattern of branch elongation and radial diameter growth can reveal the dendritic growth history of trees with the same accuracy as growthring chronologies of the trunk [Roloff, 1986]. The annual growth cycle is reflected in the branching pattern of trees and will finally be represented in the skeleton. Furthermore, the dendrochronological patterns are closely correlated to other structural characteristics of tree canopies like leaf or woody biomass [Niklas, 1994]. Allometric equations were established on this basis for many tree species in order to derive the amount of woody biomass [Bartelink, 1997], leaf biomass, [Burger, 1945] or distribution of leaf biomass in space, [Fleck, 2002], from more easily measured features such as trunk or branch diameters. Crown diameter and tree height are the easiest structural characteristics to measure. 3D-canopy light modeling depends on such spatial information as the distribution of leaves and branches and is the key to understand a number of physiological processes in the canopy that express the vitality of trees [Fleck et al., 2004].

From a remote sensing viewpoint, the automated assessment of branch dimensions in the canopy is unprecedented. Terrestrial laser scanners measure thousands of distances per second between the instrument and its surroundings at uniform horizontal and vertical angles [Shan and Todd, 2008] in order to yield a high-resolution 3D point cloud. Thus, terrestrial LIDAR enables the measurement of the complete three-dimensional structure of the branching system. This branching information can be made available to modelers in biology and forestry. An automated evaluation procedure would make it possible to overcome tedious or inaccurate measurement procedures.

The study of unorganized point clouds as an object representation and the possible information to be extracted from point clouds is an area of active research. Although the majority of research has focused on the extraction of surface properties from the point cloud, e.g. [Pfeifer et al., 2004] and [Henning and Radtke, 2008], this thesis describes a new method to reveal the branching information using the example of leafless apple trees. The fully automatic approach presented here does not depend on species information, such as allometric relationships. Obtaining the branching system from unorganized point clouds (Figure 6.1) can help in various point cloud applications. The application described here is the extraction of the branch length and diameter from laser-scanned orchard trees. The SkelTre-skeleton used in this research represents the trees branching system as an oriented graph. Such a graph consists of vertices which are connected by edges. Every vertex corresponds to a distinct part of the point cloud and is embedded into the center of the corresponding point cloud part. The edges are straight connections between the embedded vertices. The skeleton extraction from a

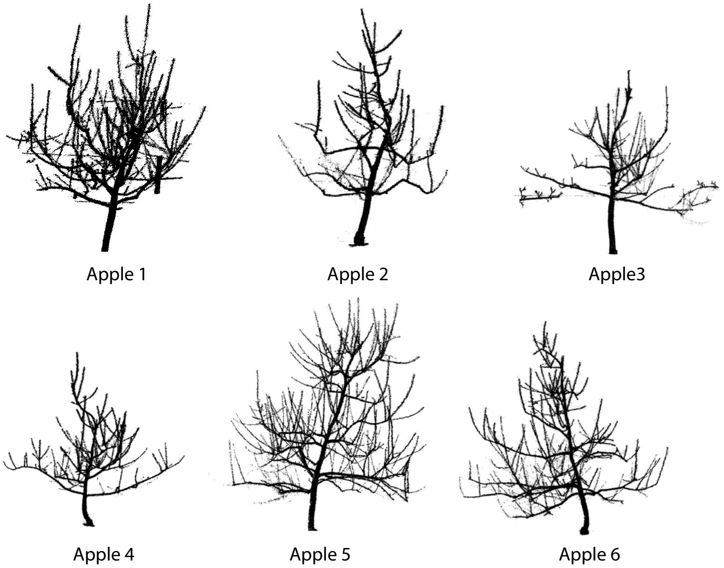


Figure 6.1: Registered 3D-point clouds of the 6 investigated apple trees with manually measured trunk diameters of 7.3cm, 6.7cm, 3.9cm, 5.9cm, 8.1cm, and 7.4cm.

point cloud has to meet several algorithmic requirements and their attributes, such as centeredness, topological correctness, and robustness to noise. These challenges are described in more detail in Chapter 2. This section is heavily based on [Bucksch and Fleck, 2009] and [Bucksch and Fleck, 2010].

### 6.1.2 Data collection and preprocessing

**Study area**

The study was mainly conducted in apple orchards of the Annapolis Valley, Nova Scotia, Canada close to the city of Kentville (45°439” North, 64°2945” West). The six investigated apple trees (*Malus x domestica Borkh. ’Honeycrisp’*) were located in two orchards that belong to the test sites of the Atlantic Food and Horticulture Research Centre, [Fleck et al., 2010]. Three apple trees grew on a *trellis system* consisting of wires to support the tree in its growth. The other three trees stood as single trees in rows. The orientation of the rows is from North to South with a tree spacing of 3m within each row and a spacing of 5m between rows. Trees of comparable height were located next to the investigated trees. The manually measured trunk diameters ranged from 3.9cm to 8.1cm. The tree height of the six apple trees varied from 1.27m to 3.03m.

### Field measurements

Each tree was scanned in March 2006 with the Imager 5003 of Zoller+Fröhlich from four sides (approximately North-East, South-East, South-West, North-West) at a distance of about 4m to the trunk. The laser scanner was placed at different heights above the ground (between 1m and 2.3m) in order to maximize coverage of the measured tree surface. The scanner resolution was set to High, which is equal to a horizontal and vertical angular step width of 0.036 degrees and results in 10.000 pixels resolution for 360 degrees. Tree branches were identified as elongated woody element with a minimum diameter of 3mm, inserting at a ramification point (node) on another, usually thicker branch or trunk element. The branches of each tree were numbered for reconstructing the branch hierarchy and their length was measured following the elongation direction of the branch. The diameter of each branch was measured at its base and tip, about 1cm before the node or end bud. The diameters of branches were measured with a caliper in two directions and averaged. If both diameter measurements were more than 1mm apart, a third diameter measurement was taken and the average of three measurements was taken. Branch diameters thicker than 5cm were derived from circumference measurements with a meter tape assuming the trunk or main branch to have a circular cross-section.

### Data processing

Registration of the scans was done with the NEPTAN-based registration algorithm in Z+F Laser Control, [Zoller und Fröhlich, 2009], based on 14-18 artificial targets that were placed on the ground and fixed to ladders at a height of about 2m in order to achieve a homogeneous distribution of tie points common to multiple scans. The 3D-point cloud was transferred to the software CYCLONE, [Leica Geosystems, 2009], and subsamples representing a single tree were isolated, [Fleck et al., 2007]. Skeletonization of each 3D-point cloud was performed with the SkelTre algorithm described in Chapter 4. Here we use the SkelTre attribute of centeredness to derive the diameter and the preserved topology to obtain a segmentation into branches and their hierarchical order.

One of the major problems with laser scanned trees is, that the youngest branches are strongly undersampled, [Bucksch et al., 2009a]. Furthermore, at higher crown densities, the amount of occlusion effects increases, leading to gaps in the point cloud because of insufficient coverage of the tree surface. The increased noise leads to the fact that some especially smaller branches may not be skeletonized. For details on this particular skeletonization algorithm, the reader is referred to Chapter 4 and Chapter 5.

## 6.1.3 Methods for tree parameter extraction

### Branch length estimation

The output of the skeletonization process is a graph, consisting of vertices connected by edges centered within the tree. Estimation of branch length requires a graph-splitting procedure (Figure 6.2) to segment the skeleton graph into subgraphs representing a single branch.

The skeleton graph allows navigation through the tree point cloud. At every vertex with more than two incident edges (marked red in Figure 6.2) the graph has to be split into the currently followed branch and newly starting branches. By tracing the graph from the trunk base, we can identify the edge $a$ (Figure 6.2) reaching a branching point. The incident edge $b$ forming the angle closest to $180\,^{\circ}$between $a$ and $b$ is selected to continue tracing (red subgraph in Figure 6.2). All other incident edges are marked as branch bases from which a new trace can be started. This procedure also provides the branch hierarchy as an output. The skeleton graph is geometrically embedded into the tree point cloud and the Euclidean length of all edges in one trace is used as the branch length.

From Chapter 4 we know that the vertices of the graph are embedded into the center of gravity of a point cloud part. This means that the angles between the edges incident to a branching point are not necessarily the angles shown in the ideal sketch of Figure 6.2. Fortunately we can create an angle close to the ideal sketch by placing the vertex representing the branching point temporarily at another location. This location is the last traced vertex before the branching point. The temporary placement is also shown in Figure 6.3. The example shows clearly that the placing is just temporary for segmentation purposes. For the length and diameter estimation the original branching point is needed, because the temporary placed vertex will affect the centeredness and will affect the length estimation for one branch. The unwanted effect in the branch length estimation would be, that e.g. the thickness of the trunk is added to the branch length.

The segmentation does not incorporate diameter information as decision criterion. The diameter is to be estimated in a second step. Either the diameter is used as a decision criterion to follow a branch along the skeleton or the angle at branching points of the skeleton. If one of the two would be known perfectly, then it would be preferable to segment incorporating both. But both are just estimations, which do contain errors. Every wrongly made decision in the tracing results in a distorted order of branches. Here, we take the angle to follow a branch and estimate the diameter at a separate step.

The output of the described procedure is a segmentation of the tree in its branches. Every branch is represented as a collection of connected line segments having no branching points.

### HARPER - Branch diameter estimation

One property of the SkelTre-skeletonization procedure is that it maintains the relationship between every vertex in the skeleton graph to a set of points $p_i$ in the point cloud. As the skeleton graph is assumed to be centered in the point cloud (Figure 6.2), the point distances of all points $p_i$ to the skeleton represent the radius of the branch. Because a $p_i$ has a relation to a vertex and not to an edge, we calculate the distances of all points $p_i$ to all incident edges of the corresponding vertex. Recall, that the maximum number of incident edges in the segmented tree is two for every graph representing

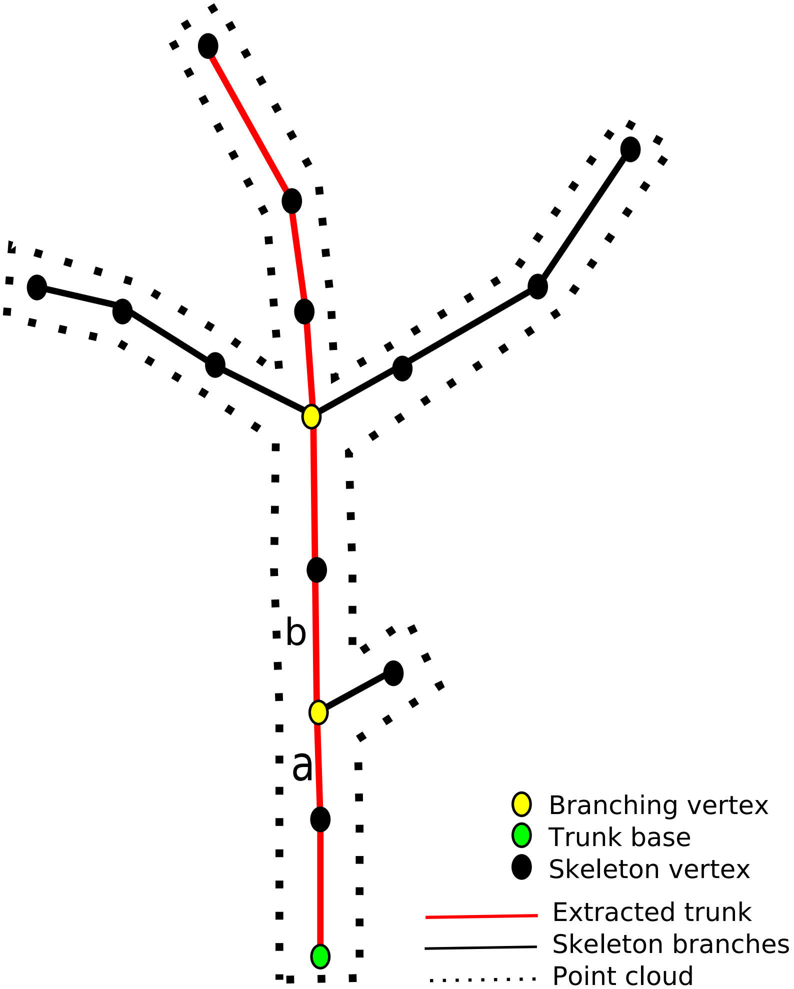


Figure 6.2: Ideal principle of the branch splitting procedure. The red subgraph is the extracted branch from the trunk base. Ordinary skeleton graph vertices are marked in black, the trunk base vertex in green and branching vertices are shown in yellow. The edge *a* is an incoming edge and edge *b* is an outgoing edge of a branching vertex in the direction from the trunk base to the branching vertex. The skeleton graph is centered within the dotted point cloud.

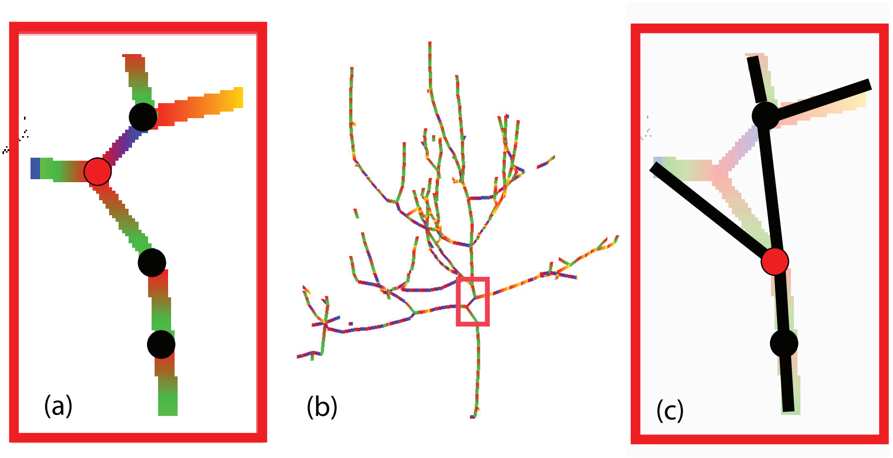


Figure 6.3: (a) a magnification of the centered skeleton part indicated in the skeleton shown in (b). The skeleton shown in (b) is the previously used skeleton of the test tree in Figure 4.4. (c) shows the temporary placement of the branching vertex.

a branch. The smallest distance of the point to one of the edges is used for further processing. After calculating the distances of all $p_i$, the histogram of distances to the skeleton is calculated, (Figure 6.4). The bin containing the median of all distances, so-called *median bin*, (red line in Figure 6.4) was chosen to avoid the influences of noise on small branches as shown in Figure 1.5 and Figure 1.6, resulting in huge distances to the skeleton. Starting from the median bin, the peak in the histogram closest to the median-bin is selected as a so-called *reference bin* for the branch radius. The average value of the reference bin is taken as the radius of the branch.

**Data preparation:** The diameter measurements from the manual and automated methods were sorted in ascending order. This sorting enabled a one-to-one comparison of the measured values, because the manually measured tree hierarchy differed from the one measured automatically. Furthermore, the manual measurement contained more measured branches than the automatically estimated branches, because branches smaller than 3mm in diameter are not captured by the laser scanner. Finer branches are strongly undersampled and result in individual points representing the entire width of a twig, as discussed on the test tree in Chapter 5 and introduced as a problem in Chapter 2. This makes it impossible to extract a diameter from it. Due to this limitation, the smaller branches from the manual measurement were removed to assure two equally sized datasets. It should be stated that the length remained extractable because the length is represented independently from the diameter as explained above and in Chapter 2. To compare the manual measurements to the automatic estimations, a linear regression of the manual measurement and the automatic estimation was calculated for all six trees in addition to compare the histogram.

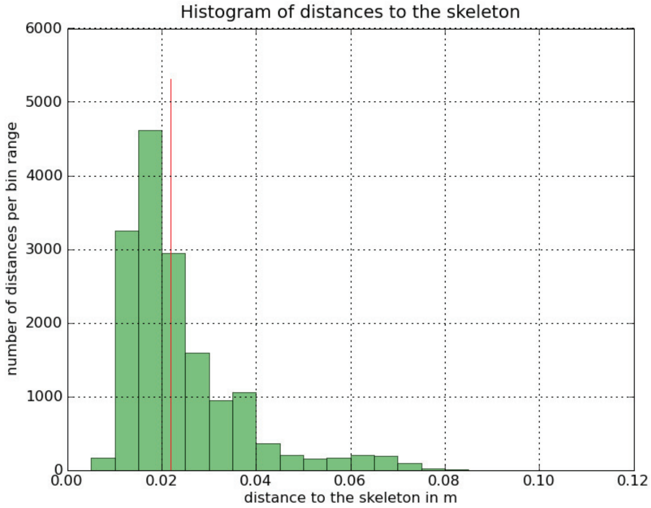


Figure 6.4: The distances of points $p_i$ of the 3D-point cloud to the skeleton were evaluated as bin-counts in a step width of 0.005m. The median of the point distances (red line) determined the reference bin for diameter estimation. The chosen reference bin in this case is the peak on the left of the red line

| Tree | Overall length of the skeleton | Overall length of the branches from the field data | Overall length of branches with extractable diameter and ratio |
|---|---|---|---|
| Apple 1 | 95.9 m | 68.1 m | 57.3 m (60%) |
| Apple 2 | 47.4 m | 41.7 m | 35.3 m (74%) |
| Apple 3 | 40.9 m | 37.9 m | 26.8 m (66%) |
| Apple 4 | 52.3 m | 53.9 m | 44.3 m (85%) |
| Apple 5 | 128.4 m | 122.8 m | 104.6 m (81%) |
| Apple 6 | 111.8 m | 103.4 m | 75.1 m (67%) |

Table 6.1: Comparison of the overall extracted length between skeleton and field measurement and the overall extracted length of branches where a diameter could be obtained. The ratio in the last column is the ratio between column 4 and column 2.

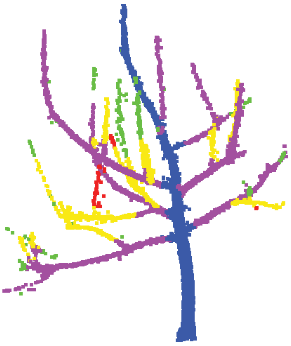


Figure 6.5: The test tree segmented into its branching hierarchy. Every color corresponds one hierarchy level. Green indicates a unconnected skeleton part.

### 6.1.4 Results and Discussion

In this section the results of the six automatically analyzed leafless apple trees, as described in Paragraph 6.1.2, are discussed.

#### Skeletons and branching hierarchy

SkelTre showed good stability to gaps and robustness to noise in the point cloud. Where gaps in the measured data occur, the algorithm still detected the two parts of a branch on both sides of the gap, as shown in more detail later on in the segmentation. This resulted in the retrieval of a higher number of branch segments than we measured by hand. Small artifacts were sometimes observed at the trunk base due to parts of surrounding ground elements (grass, moss or soil) represented in the 3D-point cloud. Figure 6.5 shows the resulting segmentation of the branches using the previously described branch tracing on the test tree used in the previous chapters. Figure 6.5 shows that the resulting derived segmentation from the SkelTre-Skeleton includes the branching hierarchy of the test tree. Note here that the green parts indicate branches which could not be connected, because of undersampling and occlusion, resulting in smaller branch segments.

#### Branch length

The automatically extracted branch lengths were compared to the hand measurements based on frequency distributions of the total amount of branches of a tree. The branch length was categorized in length classes of 5cm from 0 to the maximum occurring branch length of each tree. The results for the six apple trees are shown in Figures 6.6: While the algorithm detected a much higher number of small segments (classes up to 5cm and up to 10cm) and did recognize a few longer branches that were measured as separate entities in the hand measurements, the hand and SkelTre measurements were well correlated in the range between 20cm and 65cm.

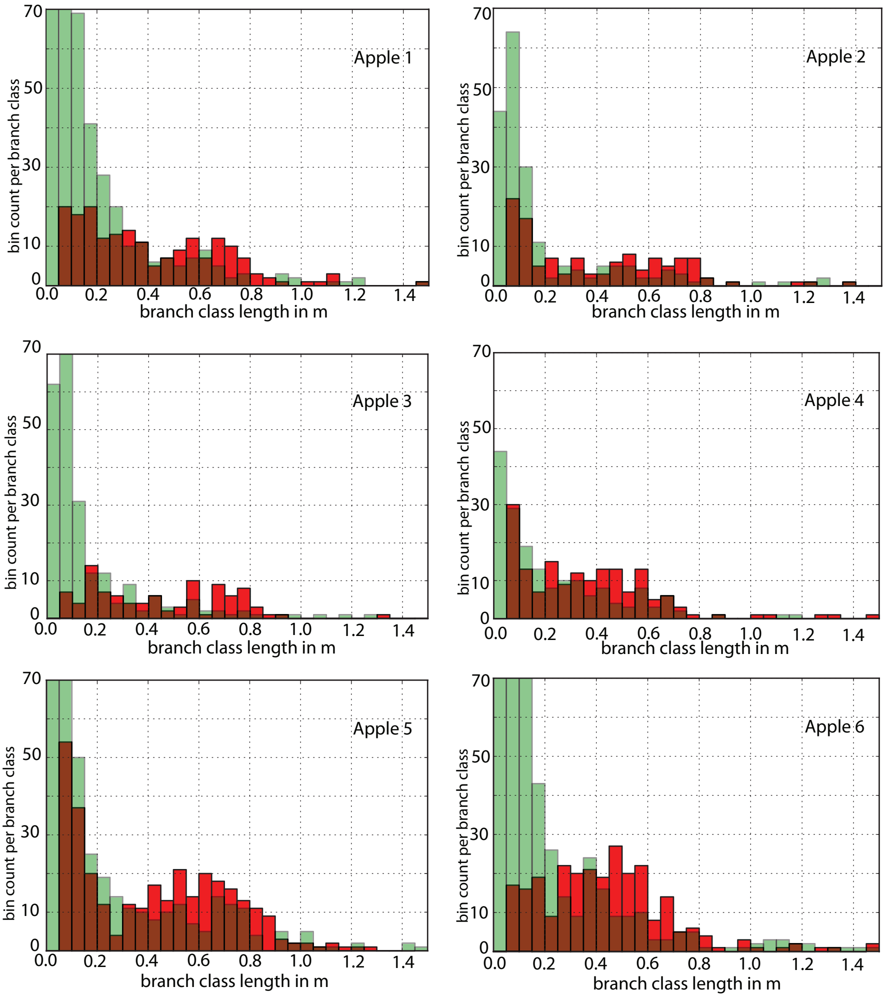


Figure 6.6: Frequency distributions of automatically detected (green) and hand measured (red) branch lengths. The histograms are binned in 0.05m bins per branch class.

| Tree | $r^2$ Length | $r^2$ Diameter |
|---|---|---|
| Apple 1 | 0.78 | 0.92 |
| Apple 2 | 0.41 | 0.95 |
| Apple 3 | 0.64 | 0.98 |
| Apple 4 | 0.77 | 0.99 |
| Apple 5 | 0.72 | 0.98 |
| Apple 6 | 0.62 | 0.99 |

Table 6.2: Results of the canopy analysis of six validation trees. The similarity of the frequency distributions is given as the correlation coefficient $r^2$ between the histograms of the field data and the automatic data.

The number of branches in the hand-measured branch-length classes was on average 49,3% of the branch number in the associated branch-length class of the automatically detected branches. Since this percentage did not substantially vary over branch-length classes longer than 10cm, all branch-length classes appeared to be similarly affected by gaps in the 3D-laserscanner data. The overall length of the skeleton and the manually measured length in the field differed not substantially except for Apple 1. The point cloud of Apple 1 contains some parts of the trellis system, which could not be filtered out and contribute therefore to the overall skeleton length. On average the the field measured length deviated 7.2% from the automatically extracted length for all trees except Apple 1. Remember here, that the chosen trees are orchard trees containing a high proportion of branches with diameters smaller than 1 cm, unlike bigger trees in a forest, where longer and wider branches occur.

### Branch diameter

For the six candidate trees the frequency distributions of field and automatic branch diameter measurements were calculated, as shown in Figure 6.7. We used a bin-size 0.005m to estimate the diameter with the HARPER-method. A high similarity of the histogram shape could be observed and assessed by linear regression (Table 6.2). A Chi-squared test to evaluate the good results of the histograms showed that there is no significant difference between the frequency distributions of field measurements and the automatic estimations. For all trees correlation coefficients above 0.9 for the diameter could be achieved (Table 6.2).

We noticed differences from the manual field measurements of up to 2 cm due to bulges not considered in the manual measurement. The field measurements were partly made with a measuring tape, which measures the convex hull of a branch, while the automatic procedure measures the smallest distance to the skeleton of the data points obtained from the hull and selects a suitable bin close to the diameter. These results are comparable to the results found in [Henning and Radtke, 2006], who evaluated the measurement error of trunk cross sections in a forest using a terrestrial laser scanner. They observed errors in the order of 1cm to 2 cm in the diameter measurements. The better correlations found for the tree diameter compared to the length measurements is explained by the differences induced by different approaches for branch decomposition between field measurement and automatic estimation. The results here show that the frequency distribution of diameters is robust to branch hierarchy errors resulting from the branch segmentation process.

### 6.1.5 Conclusion

This section presents a new approach for extracting the branching architecture from leafless apple trees. This approach is based on a skeleton extraction procedure based on terrestrial laser scan data. The similarity between the frequency distributions of the field measurement and the automatic estimations was assessed by a linear regression. On selected examples we showed that high correlation between frequency distributions of manual validation measurements and automatically extracted branch length detection is achievable, although problems with gaps in the point cloud data were obvious: The frequency distributions of the length estimations for lengths above 5cm correlated with coefficients of 0.41 to 0.78, while the diameters, where the effect of gaps does not directly influence the correlation, showed much better correlation. The frequency distributions of the diameter estimations show a similar shape for diameters bigger than 0.5 cm. The correlation coefficients of all 6 trees are above 0.9, and show that the branch segmentation has limited influence on the diameter distributions. The denser the canopy structure of trees, the more gaps are to be expected in the scanned data, a problem which remains to be solved in the algorithmic calculation.

Skeletonization algorithms such as the proposed SkelTre method provide a basis for an adequate gap-filling strategy in 3D-point clouds with a high degree of occlusion. Furthermore, a strategy is needed to encode uniquely the branching of the tree in the field. This is concluded from the result that the diameters show much better correlation than the length estimations. An algorithm derives the segmentation into branches based on defined rules. A human measuring in the field decides subjectively where a branch begins. For example small differences in the branch origin ,e.g. the first two branches from the bottom on the test tree trunk, may be ordered differently. Furthermore, the amount of branches represented in the data and measured in the field varies, because thin branches are not captured by the laser scanner. With the given validation data, containing only manually measured length and order of branches, it was not possible to derive parameter values for individual branches. However, in the next section we will validate the diameter on airborne data on a predefined location.

The overall length of the automatically extracted skeleton and the sum of the branch lengths as obtained by the field measurements differed on average by only 7.5%. Diameters could be extracted for 72% of the overall skeleton length on average. The loss of extractable diameters compared to the extracted skeleton length is explained by the strong undersampling of the finer branches. The entire extraction process of branch length and diameters was carried out without allometric relationships.

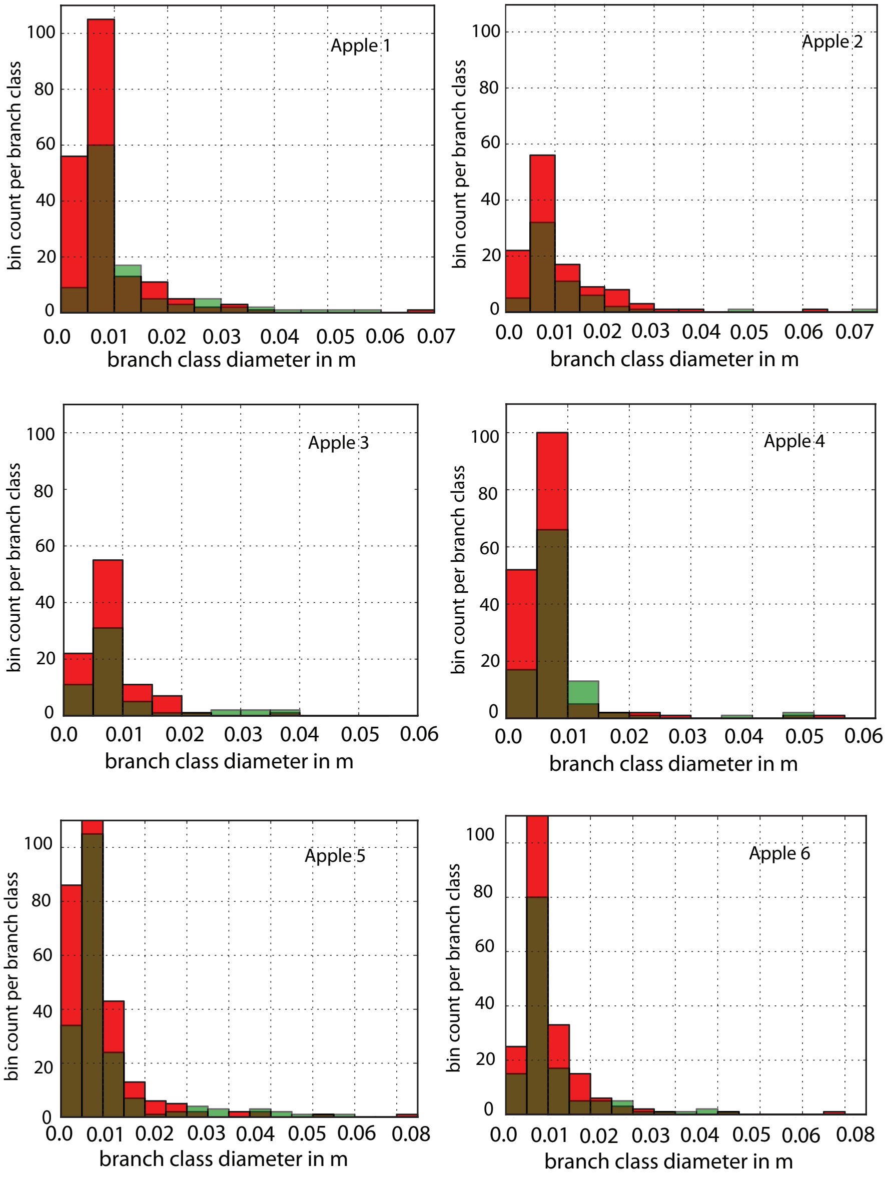


Figure 6.7: Frequency distributions of automatically detected (green) and hand measured (red) branch diameters. The histograms are binned in 0.005m bins per branch class.

## 6.2 Delineation and stem diameter estimation from Airborne Laser Altimetry

### 6.2.1 Introduction

Skeletons determined from airborne data can be used in several application domains. In forestry the diameter at breast height (stem diameter at 1.3m height) of individual trees is an important parameter to estimate other parameters like the leaf area index [Jonckheere et al., 2004] or to obtain insight in the carbon water relations in forests [Köstner et al., 2004]. Furthermore, the diameter at breast height is often used for hydrological applications, because standing trees influence the water flow in flooded areas, [Straatsma and Baptist, 2008]. Recent high density LiDAR data give a new possibility to measure the diameter at breast height of single standing trees directly. In [Fugro Aerial Mapping, 2008], it is shown that the FLI-MAP 400 system obtains high density LiDAR data with approximately 50 points per square meter. Successful estimation methods to calculate the stem diameter directly with the FLI-MAP data, rather then making a species dependent estimation, as described in [Tomokaki, 2005], are not known until now. After describing the FLI-MAP 400 system as an example of a state-of-the-art airborne platform, an overview of current species dependent estimation methods of diameters at breast height is given. After that the species independent retrieval is shown in three steps:

1. Delineation of individual trees;
2. Skeletonisation of single trees, and
3. Histogram analysis of the point distances to the skeleton.

An extensive result section shows first the possibilities on two test scenarios and secondly on a delineated forest. The first test scenario is the extraction of the diameter at breast height from an artificial data set which simulates an ideal noise-free FLI-MAP 400 system on a group of 18 trees. The second scenario includes four test cases which are delineated manually. Both scenarios avoid the influence of a delineation algorithm. Secondly, a whole forest patch is delineated automatically with the method described in [Rahman and Gorte, 2009], and diameters are extracted. This section is strongly based on the paper [Bucksch et al., 2009b]

### 6.2.2 FLI-MAP 400

The FLI-MAP 400 System, which is the follow up of the original FLI-MAP System, [Fugro Aerial Mapping, 2008], is an airborne LiDAR system that is able to obtain approximately 50 height points per $m^2$ when flying at 100m above the ground at 20 m/s speed (see Table 6.3). The FLI-MAP 400 System is designed for acquisition from a helicopter, that enables lower flying height compared to an airplane. The lower flight height is resulting in data sets that are 10 times denser than traditional laser altimetry.

According to [Fugro Aerial Mapping, 2008], the main characteristics of the FLI-MAP 400 System are:

1. Operating at very low altitudes (50-150 meters);
2. Four integrated photo and video cameras;
3. Double laser system to eliminate shadowing.

The two lasers in the system are reflectorless range finders, firing 150.000 laser pulses per second at a 60° angle off onto a plane perpendicular to the flight path. The reflected laser intensity also gives information on the observed target, but is provided as an uncalibrated product. In the Netherlands, airborne laser altimetry has been used for the production of a detailed elevation model of the whole country. The project, known as AHN (*Actual Height model of The Netherlands*), [Rijkswaterstaat, 2008], is developed to provide detailed information about elevation, highly demanded by water boards, provinces and the national government. A comparison between the new FLI-MAP 400 system, its predecessor FLI-MAP and systems used for obtaining the AHN, [Rijkswaterstaat, 2008], results in the values listed in Table 6.3. The FLI-MAP 400 system is one system used to obtain the data for the upcoming AHN2, [Rijkswaterstaat, 2008]. AHN2 is the successor of the AHN project and provides more dense data than the original AHN. Therefore new data processing methods exploiting the strength of the FLI-MAP 400 system will have applications at large scale.

| | FLI-MAP | FLI-MAP 400 | AHN Systems |
|---|---|---|---|
| Aircraft height | 50-150m | 50-400m | 1000m |
| Aircraft speed | 50-80 km/h | 50-80 km/h | 250 km/h |
| Pts/$m^2$ 100m height | 10-25 | ca. 50 | 1 |

Table 6.3: Comparison of three LiDAR systems.

### 6.2.3 Species dependent measurement of the diameter at breast height

Before estimating the diameter at breast height, the extraction of single trees from the data set is necessary, e.g. [Meia and Durrieu, 2004], [Naesset and Okland, 2002] and [Popescu et al., 2003]. This extraction process is commonly called delineation. Delineation of trees is for example possible by looking at the density distribution of height points. After computing the neighborhood of every datum point, the local density maxima are extracted to locate the tree tops. The assumption of a species dependent shape model of a tree supports the allocation of points belonging to one individual tree, [Rahman and Gorte, 2008]. Use of the crown size and tree height as input for a species dependent diameter estimation is described in [Andersen et al., 2006], [Buddenbaum and Seeling, 2007] and [Clark et al., 2004]. This assumes an allometric relation between crown size, tree height and stem diameter. The state of the art approach, described in [Korpela et al., 2007] to estimate the stem diameter at breast height is to estimate the crown radius first as a function of the tree height incorporating species dependent parameters. The tree height is derived from the LiDAR data. The diameter at breast height is then estimated as a function of tree height and maximum crown width also incorporating species dependent diameters. The major disadvantage of allometric relationships is, that the species has to be known, and does not allow the direct measurement of individual trees. Examples of such allometric approaches can be found in [Hyyppä et al., 2005], [Kalliovirta and Tokola, 2005], [Laasasenaho, 1982].

The approach introduced in this thesis does not require species dependent information to estimate the stem diameter. The estimation is done with the support of the SkelTre skeleton, describing the structure of the tree, as described in Chapter 4.

### 6.2.4 Methodology

#### Tree delineation and pre-filtering of the data

Delineation describes the process of extracting single trees from a given data set. Here we used the delineation method described in [Rahman et al., 2009] and [Rahman and Gorte, 2009]. In this delineation process a squared tile of the forest represented as a three-dimensional point cloud is used to determine the crown base height. The crown base height is found by extracting the level sets with the height function from the data. Every level set is then characterized by the number of points belonging to it. The last occurrence of a significant local minimum of points is taken as the crown base height. All points above the crown base height are projected onto the xy-plane. Again the local density of the projected points is calculated within the xy-plane. Every significant density maximum is taken as an initial seed point, which is assumed to be a tree top. From every seed point region growing is applied to find the minimum density in all directions of the xy-plane. The area enclosed by the minima in all directions is assumed to be the area of one delineated tree in the original data set.

A histogram is used to remove understorey vegetation from the delineated tree. The variable considered here is the distance between the data points of a single tree and a line perpendicular to the ground starting at the detected tree top. The bins of this single tree histogram are assumed to represent consecutively the components tree crown, tree stem, understorey vegetation and ground surface. The envelop of the frequency distribution marks each part of the tree histogram automatically as one of the components. The delineation process uses eight input parameters. The delineation seems to be most sensitive to the estimated minimum radius and maximum crown radius. As shown later, this delineation approach results in trees that are identified by the algorithm but could not be retrieved in the field (overdelineation) and trees which are present in the field, but not delineated properly (underdelineation).

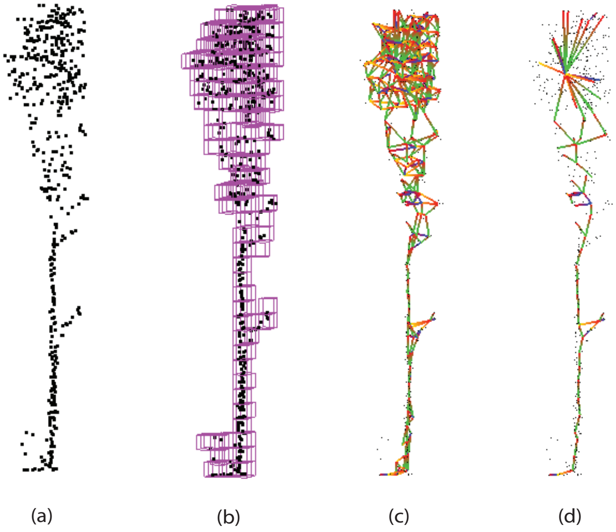


Figure 6.8: Principle of the skeletonization: (a) point cloud of a delineated tree (b) the octree organization (c) the extracted octree-graph (d) the resulting skeleton

### Skeletonisation of the trees

The delineated trees are skeletonized with the SkelTre algorithm, see Chapter 4, originally developed for terrestrial laser scan data. A single delineated tree (Figure 6.8a) is divided by an octree subdivision into small point cloud parts (Figure 6.8b). From this octree organization a graph is extracted (Figure 6.8c), which is retracted to a one-dimensional skeleton (Figure 6.8d). The SkelTre algorithm requires only the input of one user defined parameter for the purpose of airborne data. The required input parameter is the minimum cell size.

### Diameter measurement

The diameter measurement is a 3-step approach: first the stem is extracted from the skeleton, secondly a representative bin is chosen by evaluating the histogram of distances of the point cloud points to the skeleton. Thirdly, the points belonging to the selected bin are projected onto a plane with respect to the direction given by the skeleton. The *smallest diameter of an imprecise point set*, [Löffler and van Kreveld, 2007], of the projected points is used as a criterion to decide if a diameter estimation is valid.

**Stem extraction** The skeleton graph consists of vertices and edges (see Figure 6.8d). The extraction of the stem from the skeleton graph follows a simple rule. The stem is extracted by evaluating the incoming and outgoing edges of the skeleton graph's vertices, starting at the root. The root is defined as the vertex assigned to the smallest z-coordinate of the whole skeleton graph. In case of more than one outgoing edge, the edge forming an angle closest to 180 degree with the incoming edge is selected. The stem is saved as an one dimensional list of vertices, ordered by their adjacency. This strategy was already introduced in Section 6.1.

Every vertex in the skeleton subgraph belongs to a subset or segment of the original point cloud, initially derived from the octree subdivision. In order to select point cloud points relevant for obtaining the stem diameter, the skeleton is followed from the root to the stem segment that covers the diameter at breast height (1,30m above the ground). A minimum number of points along the stem is needed to evaluate the stem diameter. Here we choose a minimum of 40 points as a threshold for a sufficiently sampled stem to obtain a meaningful histogram. If the two vertices connected by the skeleton edge, which is covering 1,30m height correspond to less than 40 points in the point cloud, than the next higher segment is added to the evaluation. We keep adding higher segments, until 40 points are collected.

**Histogram evaluation** The calculation of the distances between point cloud points belonging to the stem and the skeleton graph, is done by determining for each point the distance to the skeleton edge at the height of the point.

$$d(p) = \frac{|(x_2 - x_1) \times (x_1 - p)|}{|x_2 - x_1|} \tag{6.1}$$

where $d$ denotes the distance to a line defined by two points $x_1$ and $x_2$, which are the coordinates of the two vertices of an edge. And $p$ denotes a point cloud point, such that $z(x_1) \leq z(p) \leq z(x_2)$, where $z(.)$ denotes the z-coordinate.

From the calculated distances to the skeleton, a histogram is plotted. Here it is assumed that the majority of distances is correct and results in a peak value in the histogram. The bin corresponding to the peak value is the so-called *peak bin*. For diameter evaluation the peak bin closest to the median is chosen because local blunders and starting branches can form additional unwanted peaks. The mean of the distances in the peak bin is assumed to be the estimate of the radius of the tree. In the data set evaluated here a bin size of 5cm distance was considered suited.

**Validity criterion**

The estimated diameter is evaluated to enhance the robustness of the method. Assuming that the stem is not significantly changing in its diameter along the first 4m from the root, the stem is followed upwards until a minimum of 40 points is obtained for evaluation. A simple threshold is used to determine a valid stem extraction. The motivation for such a validity criterion is given by two problems leading to potential underestimation of the diameter:

1. Shadowing may not allow the tree to be scanned from all sides, which will influence the centeredness of the skeleton, because the point cloud contains not sufficient geometric information.
2. Blunders can have a strong influence on the skeleton generated from the sparse airborne data, because they force the embedding of the skeleton outside the stem center.

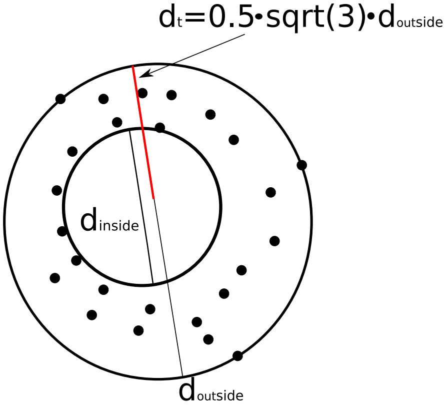


Figure 6.9: The inner diameter, $d_{inside}$ and the outer diameter, $d_{outside}$

Shadowing or blunders can lead to bad embedding of the skeleton graph. Bad embedding can lead to an unreliable centering of the skeleton and therefore leads to significantly smaller distances to the skeleton. These small distances to the skeleton result in an underestimation of the diameter. Because of this potential underestimation, we evaluate the estimated diameter $d_H$ in four steps:

1. The points used for estimating $d_H$ are projected on a plane perpendicular to the local skeleton direction.
2. An approximation $d_q$ of the outer diameter $d_{outside}$ of the projected points is determined (Figure 6.9).
3. The threshold value is computed as $d_t = \frac{1}{2} d_q \sqrt{3}$, to obtain an approximation of the inside diameter of the projected points (Figure 6.9).
4. If $d_H \leq d_t$, then $d_t$ is taken, instead of $d_H$ as an estimation of the stem diameter at breast height.

First, each vertex $v$ of the skeleton part used to estimate $d_H$ corresponds to a set of points $P_v$ in the point cloud. Starting from the vertex $v_i$ of the skeleton with the largest height value, its corresponding points $P_i$ are projected onto a plane perpendicular to the projection direction given by the edge connecting $v_i$ and $v_{i-1}$. Corresponding points $P_{i-1}$ etc. for vertices below $v_i$ are projected further similarly until the vertex $v_{i-n}$ with the smallest height value is reached. The resulting points are collected in one common plane. In practice only two projections are needed in most cases, because the minimum octree cell size needed for airborne data is often 1.5-2m.

After projecting the points, we approximate $d_{outside}$ for the outer diameter of the set $Q$ of projected points (Figure 6.9). Let $Q = \{q_1, ..., q_{|Q|}||$ be the set of all projected points. Then,

$$d_Q = \frac{1}{|Q|} \sum_{i=1}^{|Q|} max_{p \in Q} ||p - q_i||, \tag{6.2}$$

is the average of the largest distances for every point $q_i$ to the points in $Q$. $d_Q$ is an approximation of the outer diameter $d_{outside}$. Note here, that locally extreme outliers would result in a branching of the skeleton. These extreme outliers are removed by the stem extraction. Because of this the computation in Equation 6.2 may only be affected by smaller single outliers. Their influence is moreover limited by taking the average overall largest distances.

The threshold $d_t$ used here in step 3 is given in [Löffler and van Kreveld, 2007] for modeling the smallest diameter of an imprecise point set as an constant factor approximation, based on the smallest enclosing circle of a point set. Roughly, an imprecise point is modeled by a circular region to describe the possible difference between the location of a point on a surface and the measured value representing the point. This difference was already stated in Section 2.1.

### 6.2.5 Results and Validation

The validation of the method is done in 3 steps. First on a noise-free simulated test data set, which simulates the point density of the FLI-MAP 400 system. From this simulation insight in the occlusion effect in a forest, when scanned with an airborne LiDAR system, is obtained. Secondly, four representative test cases are analyzed in detail and compared to a standard cylinder fitting result. As a third step we use an automatic approach to delineate a forest sampled with the FLI-MAP 400 system, Figure 6.12, in the “Duursche Waarde” in the Netherlands, Figure 6.14. From these delineated trees the diameter is derived. All delineated trees from the test area are illustrated in Figure 6.12.

### 6.2.6 Simulated data

The data used in this thesis, [Rahman, 2011], simulates 75 points per square meter measured from the top of an artificial forest patch. This forest patch contains 18 trees with diameters ranging equally in 0.1m steps from 0.3m to 2.0m. Examples of these trees are shown in Figure 6.10. The simulation data is free of noise and incorporates only the effect of shadowing. Figure 6.11 shows the results of the 18 test trees. Insufficient point density prevented the analysis of three trees using the HARPER method on the simulation data. For the 15 remaining trees an excellent correlation coefficient of 0.97 could be obtained. The scatter of the observations with respect to the regression line is expressed by the root-mean-square error of the residuals [1] to the regression line and resulting in 0.05m. In the experiment here we used a minimum cell size of 2m to achieve the good correlation results with the SkelTre method as shown in Figure 6.11.

---

[1] more often the term standard error is associated to the root-mean-square error of the residuals in a regression analysis

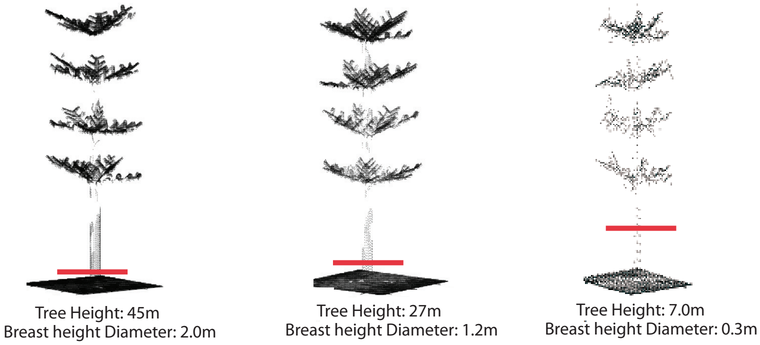

Tree Height: 45m
Breast height Diameter: 2.0m
Tree Height: 27m
Breast height Diameter: 1.2m
Tree Height: 7.0m
Breast height Diameter: 0.3m

Figure 6.10: Fli-Map 400 simulation with 75 points per square meter. 3 example trees from the simulation data set with the breast height indicated by a red line

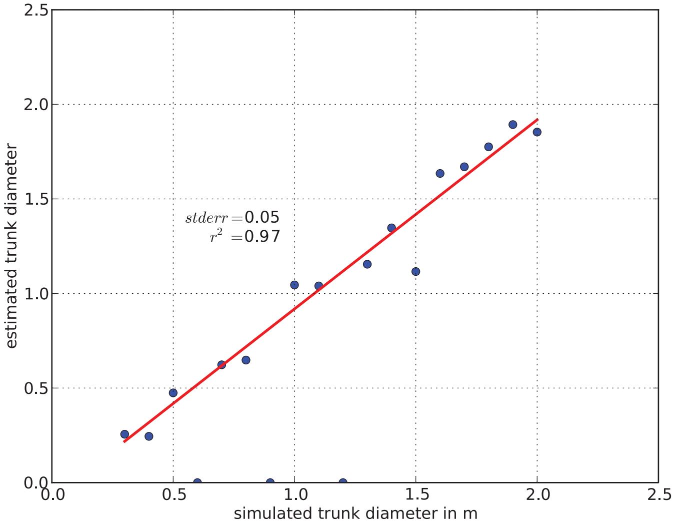


Figure 6.11: The regression analysis of the 18 test trees from the simulation data set. Three trees could not be measured and were excluded from the regression analysis.

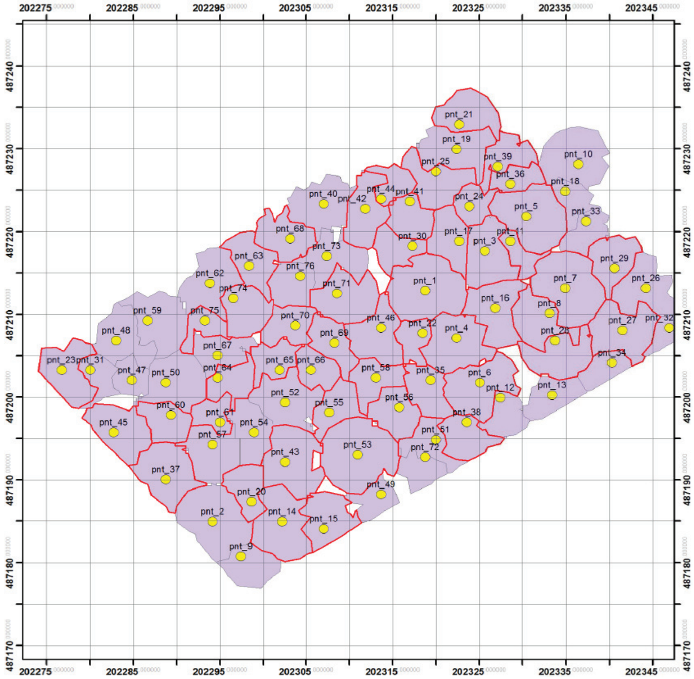


Figure 6.12: The delineated trees from the forest patch in the Dutch RD coordinate system. Note that this is the delineation result and not the actual map of the trees.

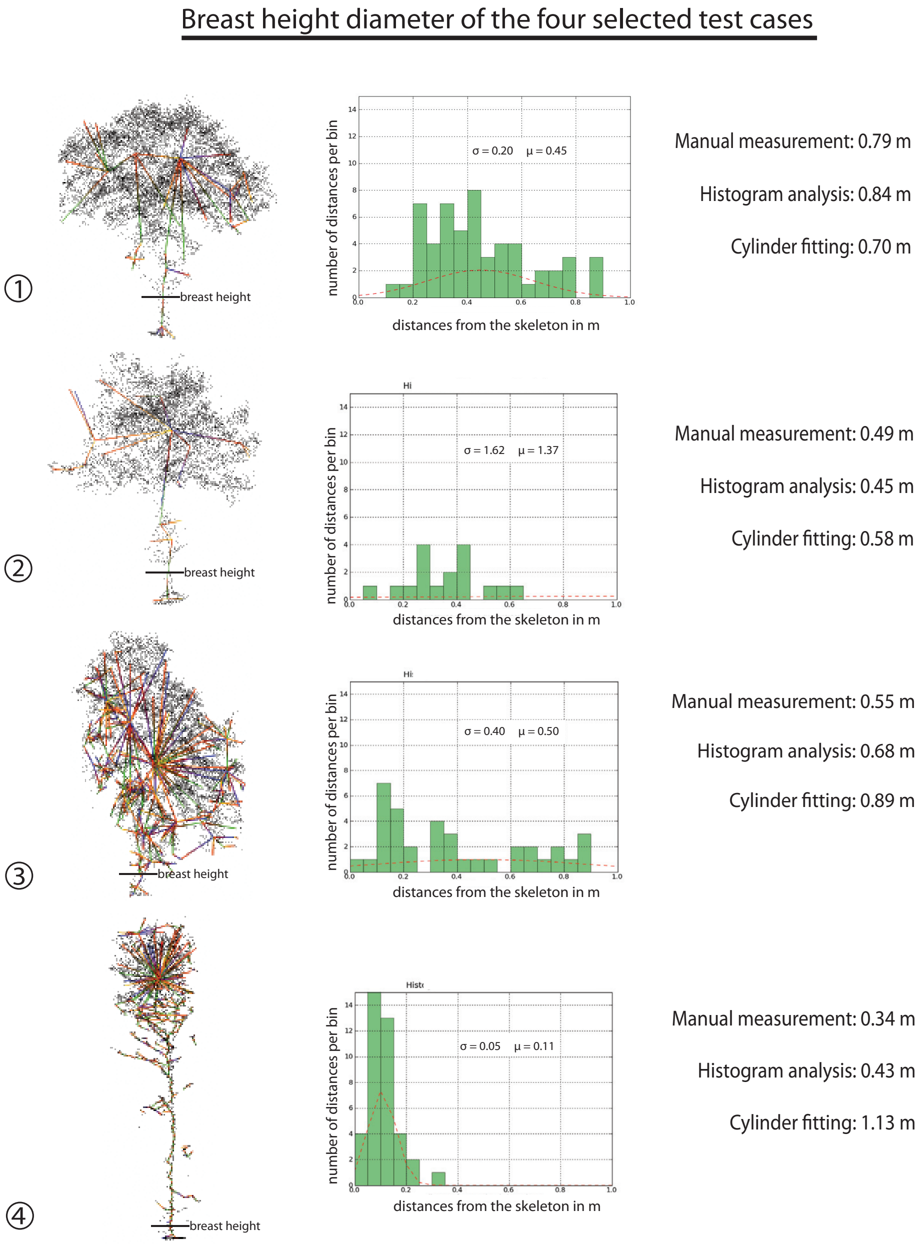


Figure 6.13: Four test cases marked at breast height. For every tree the manual measurement, the HARPER estimation and a cylinder fitting result is given as a diameter estimation: 1.) A single standing tree, 2.) a single standing tree with noise around the stem, 3.) a tree from the border of a forest with huge shadowing parts and 4.) a tree delineated from inside a forest.

### 6.2.7 Four test cases

Four test cases of individual FLI-MAP 400 sampled trees were chosen to evaluate the HARPER method (Figure 6.13). A simple single standing tree (Nr.1), a single standing tree containing noise on the stem (Nr.2), a tree on the border of a forest with huge shadowing (Nr.3) and a tree delineated within a forest (Nr.4). For every tree the point cloud with its corresponding skeleton, the extracted stem, the histogram to be analyzed and the measurement results are given. We give in total three measurements for each test case. The manual measurement as a ground truth, the result obtained by the HARPER method and the result of a fitted cylinder. The cylinder was fitted with the point cloud package Cyclone 5.7 of Leica Geosystems, [Leica Geosystems, 2009] by selecting a subset of the point cloud by visual analysis. The four test cases were delineated by hand in order to eliminate the influence of the delineation method. The first tree can be seen as a standard case giving good results. The second tree is an example where insufficient points to estimate the diameter are present. A tree suffering from huge shadowing effects (Tree 3) still gives an acceptable result, even if the skeleton extraction was far from optimal. A 13cm difference to the validation measurement was obtained. The tree from the interior of the forest (tree 4) shows a clear peak in the histogram. A huge difference is observed in the comparison between the fitted cylinder and the automatically estimated diameter. In this case the stem was not represented properly in the point cloud. By applying the introduced validity criterion an estimation of the diameter at breast height could be obtained that only deviates 10cm from the manual measurements. In all four test cases the histogram analysis performed better than the cylinder fitting with standard software.

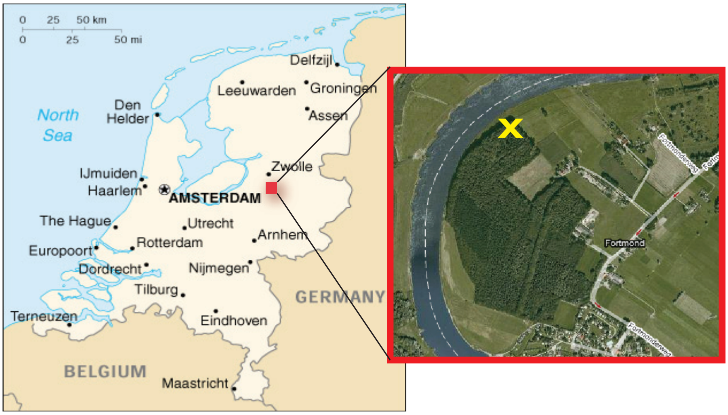


Figure 6.14: The test area in the Netherlands marked with a red box and the investigated forest patch indicated by a yellow cross.

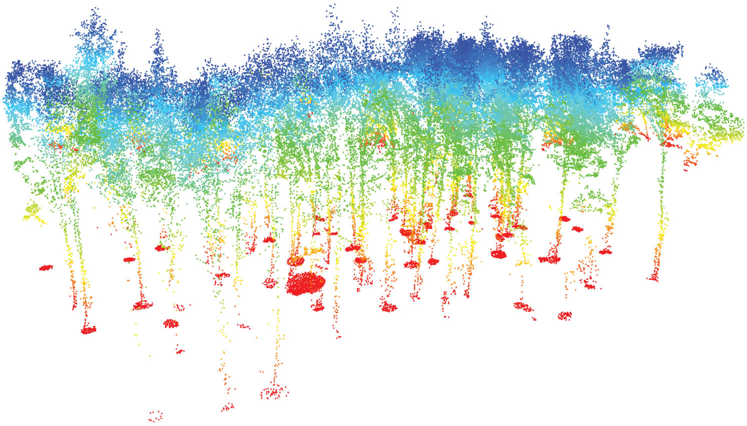


Figure 6.15: The trees delineated from the data set. Note the missing stems of some trees.

### 6.2.8 Delineated forest trees

49 trees were delineated from a FLI-MAP 400 data set in Figure 6.15. The data set was collected over a mixed forest stand containing leaf and pine trees in the “Duursche Waarde” in the Netherlands. The problems arising in this particular area are mostly connected to finding back the trees in the forest. The dense canopy does not allow precise GPS measurements to localize tree positions reliably. Furthermore, depending on the chosen input parameters of the delineation algorithm [Rahman and Gorte, 2009], [Rahman et al., 2009] trees may be wrongly classified as understorey vegetation. In the patch in Figure 6.12 a manual inspection revealed five trees that were not detected by the delineation method. One tree that was delineated from the data set was lying on the ground at the time of the validation measurement. Some trees could not be retrieved in the forest. In total it was concluded that 49 trees were usable for validation. From these 49 trees 14 had not sufficient stem points available for analysis after filtering understorey vegetation.

The diameter of the 49 trees within the selected forest patch was extracted with the HARPER method. The overall standard deviation associated with our estimation method is 0.19m. The diameters show an average deviation of 28cm from the manually obtained values. Especially the delineated trees in the interior of the forest lead to strong under estimations. Table 6.4 and 6.5 show the manual and HARPER estimations of the diameter at breast height of trees retrieved in the forest. The tree numbers are corresponding to the numbers given in Figure 6.12

| Tree ID | HARPER | Manual |
|---|---|---|
| 1 | 43cm | 39cm |
| 2 | n.a. | 52cm |
| 3 | 45cm | 37cm |
| 4 | 24cm | 36cm |
| 5 | 35cm | 43cm |
| 7 | 35cm | 47cm |
| 8 | 64cm | 47cm |
| 9 | 66cm | 48cm |
| 10 | 66cm | 56cm |
| 11 | 28cm | 38cm |
| 14 | 24cm | 40cm |
| 15 | 20cm | 43cm |
| 16 | 44cm | 37cm |
| 17 | 32cm | 45cm |
| 18 | n.a. | 37cm |
| 19 | n.a. | 172cm |
| 20 | 14cm | 40cm |
| 21 | 39cm | 37cm |
| 22 | 56cm | 33cm |
| 23 | n.a. | 69cm |
| 24 | 16cm | 30cm |
| 25 | 36cm | 52cm |
| 28 | 28cm | 46cm |
| 30 | 17cm | 51cm |

Table 6.4: Comparison between the HARPER diameter estimation and the manually measured diameter in the field for the trees 1-30. The tree numbers are correspond to the numbers in the tree map of Figure 6.12. n.a.= not available and denotes a tree that was not retrieved in the delineation.

| Tree ID | HARPER in cm | Manual in cm |
|---|---|---|
| 36 | 19cm | 49cm |
| 37 | 16cm | 44cm |
| 38 | n.a. | 34cm |
| 39 | 36cm | 2 Stems: 60cm and 55cm |
| 40 | n.a. | 39cm |
| 41 | 83cm | 32cm |
| 42 | n.a. | 50cm |
| 45 | 20cm | 45cm |
| 46 | 47cm | 19cm |
| 50 | n.a. | 37cm |
| 53 | 25cm | 46cm |
| 55 | 65cm | 39cm |
| 57 | n.a. | 54cm |
| 59 | n.a. | 46cm |
| 60 | 25cm | 39cm |
| 61 | n.a. | 34cm |
| 62 | n.a. | 54cm |
| 63 | n.a. | 47cm |
| 65 | 20cm | 45cm |
| 68 | 32cm | 38cm |
| 69 | 67cm | 34cm |
| 70 | 30cm | 28cm |
| 71 | 15cm | 28cm |
| 74 | 120cm | 41cm |
| 75 | n.a. | 25cm |

Table 6.5: Comparison between the HARPER diameter estimation and the manually measured diameter in the field for the trees 31-75. The tree numbers correspond to the numbers in the tree map of Figure 6.12. n.a.= not available and denotes a tree that was not retrieved in the delineation.

### 6.2.9 Conclusion

In this section we presented a proof of concept for a method to automatically extract the diameter at breast height from trees sampled by the high density airborne laser data system FLI-MAP 400. On a simulated data set of 18 trees and on four individual scanned test trees the HARPER method was demonstrated. Furthermore, this HARPER method was validated against manual measurements of 49 trees in a forest patch in the “Duursche Waarde” in the Netherlands. The results indicate strongly, that there is need for new delineation methods, in order to make the HARPER method feasible for complex forests. Several trees were found, which contain no stem data.

Because of the difficulty to find back individual trees in a complex forest, it was not possible to get a large and reliable validation data set based on a delineation. Results on the simulated trees and on the four individual trees are however promising. For further validation, more data of isolated trees should be collected. Two aspects have to be investigated in more detail, the influence of the delineation process and the influence of the FLI-MAP 400 system. Nevertheless, this case study gives enough evidence to conclude that skeletonization of trees is feasible for high resolution airborne data. It is predictable that denser data will be available in future, which positively influences the performance of already known delineation algorithms and the quality of the skeletonization. At the time of finishing this thesis I already saw data sets of up to 120 points per square meter. Here we used just 75 points per square meter.

## 6.3 Summary

In this chapter two applications of the SkelTre method were presented using the newly developed HARPER method to extract a diameter from a point cloud supported by a SkelTre-skeleton.

In Section 6.1 an application to automatically measured orchard trees was presented. This application showed the possibility to extract the tree parameters branch length and diameter from terrestrial laser data. The trees used in this research contained branch diameters smaller than 1cm. Because of that, this application is an excellent example of the data handling problems described in Chapter 2. The main problems found in this application are the difference between manually measured branch hierarchy in the field and the automatically extracted branching hierarchy. On strongly undersampled fine branches the skeleton becomes disconnected. This disconnection results in a large amount of branches shorter than 2cm. An assessment of the individual values for length and diameter of branches was not possible with the given validation data. To derive 1-1 correspondences a new experiment is necessary, where markers are placed in a less complex tree crown. A good experiment would be an experiment in a greenhouse, without human intervention changing the form of the tree (e.g. branch pruning) and free of weather influences. Nevertheless, a validation of the diameter estimation on a predefined location on the tree trunk was possible on the simulated airborne data set.

The second application example in Section 6.2 is the estimation of the diameter at breast height from high density airborne data. This application is aiming strictly on future possibilities of skeletonization methods. It was shown on simulated and real world test cases that it is possible to extract the diameter at breast height from a sparse airborne laser point cloud, although new methods will be needed to delineate the trees from each other to obtain reliable results on a large scale.

Both applications do not require species-dependent allometric relationships.

# 7

# Conclusions

In this concluding chapter we first review the results of this thesis on the basis of the objectives in Chapter 1. After that we give an outlook towards possible applications and extensions of the methodology developed in this thesis.

## 7.1 Summary of the results

The core result of this thesis is the development of the SkelTre-skeleton algorithm, that extracts a skeletal structure from point cloud data. The developed skeleton algorithm is especially designed to extract a skeleton from the imperfect data obtained by scanning devices such as terrestrial laser scanners. The idea for developing such an algorithm finds its origin in the wish to analyze laser scans of trees. Therefore, the main objective reads as: *Is it possible to automatically reveal the hidden structure of botanical trees from laser scan data?*. Answer to this main objective was given by investigating the sub-objectives, as given in Section 1.2. The objectives are grouped into objectives regarding the methodology and objectives aiming on applications. First we discuss the methodology related expected results.

*1.) The extracted skeleton is centered within the represented object and represents the order of object parts correctly, to enable the extraction of tree parameters from point clouds of botanical trees.*

In Chapter 4 a strategy to embed the extracted skeleton into a point cloud was given. It was shown in Chapter 5 on a point cloud of a tree that this strategy performs better than a previous algorithm of the same class. From the elongation direction of the object the branching points of the object are derived and encoded in the SkelTre-Skeleton, as stated in the attributes for topological preservation in Chapter 2. The resulting skeleton is a graph embedded into the point cloud, where every vertex of the graph is related to a part of the point cloud. The success of this embedding strategy is also shown in practice on several examples of scanner obtained point clouds in Chapter 5. These point clouds represent trees and objects that significantly differ from the form of a tree. We investigated the attributes centeredness and transformation mentioned in Chapter 2.

*2.) The skeleton has to be extracted from an unorganized point cloud, which is the given input from a registered laser scan.*

Through the whole thesis no assumption was made on a predefined order of the points. Because of that the algorithm is capable to process unorganized point cloud data. The new skeletonization method was demonstrated and analyzed on 6 practical examples of object sampled as imperfect point cloud in Chapter 5. These examples were discussed in detail in relation with the difficulties arising with the imperfectness of the data and the object form. These difficulties were described in Chapter 2 as the handling of imperfect data requirement.

*3.) A skeleton extraction process must be robust to noise, varying point density and undersampling as produced by measuring instruments.*

An octree was used to subdivide the point cloud spatially into cubic cells. Investigating the point cloud per cell results in local elongation descriptions of the object. These elongations are often derivable even under the presence of noise, undersampling and varying point density. A robustness criterion was modeled into the algorithm to further enhance stability and to deal with different noise characterizations of other point cloud generating instruments in future. The data used in thesis were obtained with different scanning devices to produce the point clouds. Scanning devices such as close range scanners where used to e.g. produce the point cloud of the fertility statue in Chapter 5. Mid-range scanners like terrestrial laser scanners sampled the trees discussed in Chapter 5 and Chapter 6.1. In Chapter 6.2 airborne scanners were used to obtain data from individual trees and forests.

*4.) Laser scanning produces huge amounts of data. Therefore, the skeletonization algorithm should be fast. The algorithm speed is typically expressed by the computational complexity, which is a machine independent measure of how the algorithm scales in time by increasing the number of input points,*

In Chapter 5 it was shown that SkelTre scales as O(n) with the number of octree cells, i.e. it can deal efficiently with huge data sets. The localized concepts of SkelTre enable this particular algorithm to be parallelized, such that an adaption can be developed for super computing facilities. In practice we have shown the skeletonization on point clouds of up to 1 million points, (Chapter 5). It should be noted, that the algorithm performance depends more on how complicated the structure of the object is, rather then on the amount of input points. This measure was stated in Chapter 2 as an attribute of computational efficiency.

*5.) The algorithm should contain as few user input parameters as possible*

SkelTre contains only one input parameter, the minimum cell size, which makes the SkelTre algorithm user friendly. This input parameter allows to adapt the algorithm to different kinds of data and defines how many changes in the local elongation direction are considered on the object. Therefore the algorithm has the hierarchy attribute stated in Chapter 2 for computational efficiency.

The possible practical value of this thesis was reflected by three sub-objectives:

*1.) The applicability of the algorithm should be demonstrated on example trees. If the objectives above are achieved, the practical result is the extraction of branch lengths* *and diameters along with the branching hierarchy of a tree.*

The major benefit of using skeletons combined with terrestrial laser data is that no species information is involved in the tree parameter calculation. This possibility indicates strongly that terrestrial laser scanning has the potential to provide species independent measurements of trees. This was shown on two example applications. The first application considers estimates of branch length and diameter of fine structured orchard trees which were scanned with a terrestrial laser scanner. The second application is the breast height diameter estimation of trees using high density airborne data.

Chapter 6.1 introduced the analysis of terrestrial laser scanned orchard trees to determine branch length and diameter. In order to assess the branch length and diameter extraction, the frequency distributions for a given tree of both, manual and automatically extracted branch lengths and branch diameters were given. For both parameters, a high correlation between field and manual measurement was found. Especially the correlation for the diameter is high: above 0.9 for all investigated trees. In order to extract the branching hierarchy from a laser scanned tree, the SkelTre skeleton was used to segment the point cloud into ordered branches. The SkelTre skeleton is suitable for this segmentation task, because every vertex in the skeleton relates to a unique part of the point cloud. The very thin branches of the orchard tree considered in Section 6.1 are not sufficiently captured by a scanner. Because of that the manually derived hierarchy differs to a large extent from the automatically derived hierarchy, especially at the branch ends.

The second introduced application is obtaining the breast height diameter from high density airborne data. Obtaining the breast height diameter relies on the new HARPER method (Section 6.2) to derive the diameter from a point cloud by using the skeleton. HARPER extracts a subset of points at a defined location in the point cloud. This subset is used to estimate the diameter at 1.3m height of the trunk, the so-called breast height diameter. A point cloud obtained by simulating a high density airborne laser scan over a forest model was used in an experiment to validate the trunk diameter extraction. Between the diameters used as an input to generate the tree models and the diameter estimates of the HARPER method an excellent correlation coefficient of 0.97 was achieved. Manually delineated examples of single standing trees showed good results as well. On the manually delineated trees of real airborne laser data the estimated diameter using HARPER was better than the diameter obtained by cylinder fitting in comparison to the values obtained by manual field measurements. The added value of this experiment is that it demonstrates the future potential use of the developed methodology: the simulated and real point clouds fulfill the requirements of the currently constructed Actual Height Model of the Netherlands (AHN2). AHN2 will cover the whole of Netherlands with such airborne laser data in 2013.

*2.) Identify current limitations in the application of skeletons due to common practice or dependency on other algorithms.*

It was difficult to compare individual branch lengths as obtained from field measurements and automatic SkelTre-based estimates. Furthermore, an analysis of the individual branches was not possible, because the configuration of branches measured in the field on one hand and the extracted configuration captured by the laser scanner on the other hand show high differences. I would recommend a limited experiment on trees with thicker branches and less complex crown to validate parameter values, extracted by means of the SkelTre algorithm, such that 1-1 correspondences can be found by e.g. the use of markers.

The extraction of the breast height diameter from high density airborne data is mainly constrained by the performance of the used tree delineation algorithm. Current delineation methods have no guaranteed performance on a whole forest, because not all trees found by the delineation method were present in the forest. Also it is still not decidable how the removal of understorey vegetation influences the estimation of breast height diameter.

*3.) Test the performance on objects other then tree structures, to evaluate the generality of the introduced concepts.* A tree contains no loops and contains obvious elongation in its branches. The introduced framework in Chapter 4 does not restrict to the number of loops formed by an object. In practice we found no restrictions for the SkelTre algorithm on non-tree-structures forming up to 4 loops and an outer form with less elongation in Section 5.2.2.

By answering the objectives above I conclude that the main objective:

*Is it possible to automatically reveal the hidden structure of botanical trees from laser scan data?*

is answered with: YES.

As a result of this thesis the new SkelTre algorithm for point cloud skeletonization as described in Chapter 4 was developed. This algorithm was evaluated in Chapter 4 and Chapter 5, with respect to the desired requirements topological preservation, metric representation, handling of imperfect data and computational efficiency given in Chapter 2. This evaluation also revealed, that the shortcomings of current algorithms as introduced in Chapter 3 are improved. Furthermore, the modularity of the algorithm allows easy further development on embedding and robustness of the skeleton algorithm.

## 7.2 Outlook and future work

### 7.2.1 New methodology

In my opinion an intermediate product of the SkelTre method, the so-called octree graph, is useful for extracting a surface from a point cloud. The octree-graph (Section 4.2.2) contains ideally all information on how the object surface is placed in Euclidean space. Furthermore, SkelTre can be used to topologically constrain surface reconstruction methods acting on the point cloud as a whole to improve the surface reconstruction result.

Another basic problem in laser obtained data is the registration. I suggest strongly that SkelTre will be used to register point clouds based on the given relations between the original point cloud data and the skeleton itself. Within a typical registration process the branching points can be used as features on which the initial transformation between two point clouds can be calculated and the connections between branching points can be exploited to perform the fine registration on the initial data set. Such approaches could be useful, if small changes affect the different scans in a registration procedure. Such small changes could be caused by rain between two scans, for example, resulting in heavier and therefore sagging branches. Another aspect is that the targets placed in the surrounding tree crowns to assure good target distribution are always slightly moving because of wind. I expect that placing targets in the tree crowns is not necessary, if a skeleton is used to perform the registration.

As a recommendation for further development of this particular skeleton I would suggest to replace the octree structure by a data structure utilizing spheres instead of cubes. For example a Poisson disk sampling, [Cline et al., 2009], can be used to sample the initial point cloud. Such an subdivision of the point cloud may enable to use more than 6 principal elongation directions. Nevertheless, I expect several years of research on that before this will be solved.

### 7.2.2 Tree and forest applications

This paragraph gives an outlook of applications where SkelTre might be used in future. Straightforward is the calculation of the individual wooden tree volume based on SkelTre and the HARPER method to estimate the diameter.

Secondly I expect skeletonizations in general to be a solid basis to estimate forest parameters such as crown base height, crown density and average distance between trunks. All the named parameters characterize the analysis of the free space within a forest, which may be linked to quantities like the amount of insects in the crown, the turbulent exchanges of momentum heat, vapor and $CO_2$ or how much light is penetrating at certain locations through the crown on the ground.

SkelTre may also be used for solving the problems of the delineation of single trees from high density airborne data of trees, as introduced in Chapter 6. A methodological drawback of the delineation method used in Chapter 6.2 is that it is necessary to map the 3D data to lower dimensions like 2D density maps and 1D histograms. The use of a skeleton will give the chance to overcome this limitation and allow the direct analysis on localized structural descriptions.

Species-independent measurements of tree stands with laser scanning based on skeletons are promising. A main benefit is that the manual collection of species information in cities and forests could be largely avoided. This statement is closely related to the next statement.

The, in my opinion, scientifically most relevant, but wide open question is how much species information is coded in the tree structure. Many parameters such as $CO_2$ exchange, resilience to climate variability or dust filtering capabilities depend largely on the tree species. Furthermore, the identification of tree species can also enable the search for unknown tree species.

### 7.2.3 Other applications

A specific closely related application, where I see SkelTre already as a useful tool is the investigations of river and valley networks , [Baker et al., 1992], recorded as a point cloud. A point cloud of such a network can be derived e.g. from satellite stereo images. These networks have a very similar structure to the tree crowns used in this thesis. Such an application could greatly make use of the modularity of SkelTre. For example the robustness criterion can be used to model error behavior on different data.

Furthermore, I expect skeletons to be useful as a data structure to encode point clouds or their derived products such as 3D models in databases. Especially an application of encoding whole cities on different level of details I consider as valuable for town administrations. This is motivated by the fact that skeletons contain hierarchical information about the objects and if applied to a whole scene they contain also the relation between objects, such as houses, roads, lamp poles etc., of a city. It is already known that skeletons are excellent object classifiers, but their use as hierarchical data structure is to my best knowledge not widely spread.

New measurement instruments, such as range cameras, produce point cloud videos. These point cloud videos are a new application field for skeletons. Here especially the recognition of objects is relevant. I predict that we will have similar encodings of these 3D videos like now in MPEG4, [Richardson, 2003], which will separate the static background from the moving objects. Detected objects can be encoded, and the skeleton can be used to describe the movements of the object parts of an detected object. An example for that could be a running human or a tree in the wind.

### 7.2.4 A far away dream

If we look further up to the level of a dream within the application scope of this thesis, we can even think of a model defining a tree as a measurement instrument. I believe a lot of information is encoded in the outer form of a tree. Simply suppose that every branch is representing a certain time span. Factors like nutrition conditions, how light and weather influence its growth and therefore its form and structure. I believe that there is much information encoded in the actual structure of a tree which is relevant for our daily life and it is a matter of time until we will discover all information coded in it. And I believe we will find such ’Ecometrics’ to express what information is coded in the tree structure and that there is no need to cut a tree to derive this information.

### 7.2.5 And what about the leaves?

On a tree with leaves the first results are already promising. It seems like that SkelTre could evolve to derive a skeleton from trees with leaves (Figure 7.1).

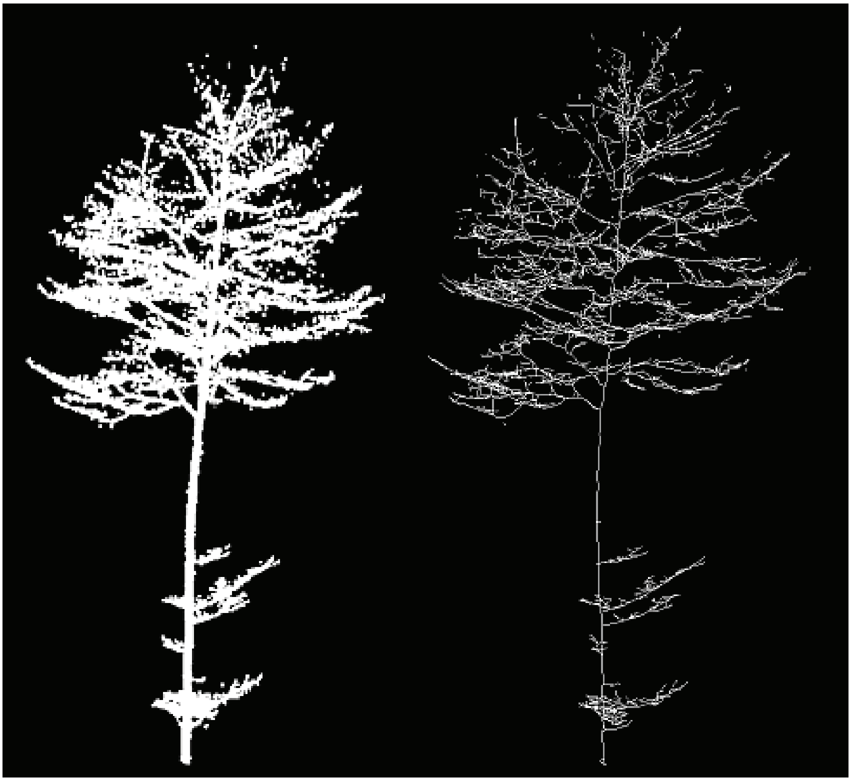


Figure 7.1: (left) point cloud of a tree with leaves and (right) the SkelTre skeleton corresponding to the point cloud on the left

# A

# Glossary and Symbols

$\oplus$ denotes the merging operation of E-Pair or V-Pair (Section 4.3) in a graph or the morphological operation erosion (Section 3.2.1).

$\ominus$ denotes the morphological operation Dilatation (Section 3.2.1).

$\times$ denotes the Cartesian graph product (Section 4.2.2).

**big-O notation** denotes the scaling of a process in time or space. Let $n$ be the length of the input, then $f(n) = O(g(n))$ means there are positive constants $c$ and $k$, such that $0 \leq f(n) \leq cg(n)$ for all $n \geq k$. The values of $c$ and $k$ must be fixed for the function $f$ and must not depend on $n$ (Section 2.6).

**CAMPINO** abbreviates **C**ollapsing **a**nd **m**erging **p**rocedures **in o**ctree-graphs and is a skeletonization method for point clouds based on graph-reduction.(Section 3.3.3).

**computational complexity** is a notion for the the scaling of an algorithm in time or space (Section 2.6). See big-O notation.

**dilatation** Let $A$ be the set of raster cells, e.g. containing point cloud points, which is modified with a set $B$. $B_x$ denotes the translation of $B$ in direction of vector $x$. $A \ominus B = \{x | B_x \subseteq A\}$. See Section 3.2.1.

**direction label** belong to edges in the octree-graph and denote a principal Cartesian direction. A label associated with an edge of the octree-graph indicates the direction of the edge with a direction vector. The labels for all 3D-directions are: Right/Left: $(\pm 1, 0, 0)$, Up/Down: $(0, \pm 1, 0)$, Front/Back: $(0, 0, \pm 1)$ (Section 4.3).

**dominant direction** indicate a vertex direction where a valid merge is possible. Let $x_1, x_2, x_3$ be the three components of $vdir(v) = (x_1, x_2, x_3)$. A direction of $x_i$ is trivial if the Cartesian entries are zero for each associated edge label. A direction $x_i$ is dominant at $v$, iff $x_i = 0$, but not trivial. A direction corresponding to a non-zero value for $x_i$ is called a non-dominant direction (Section 4.3).

**E-Pair** is a Pair of vertices to be merged by the SkelTre algorithm. An E-Pair merged E-Pair generates V-Pairs. Let $v_i$ and $v_j$ be two adjacent vertices with

$vdim(v_i) \leq vdim(v_j)$. Then $v_i$ forms an E-Pair with $v_j$ if: 1. $vdim(v_i \oplus v_j) \leq \max(vdim(v_i), vdim(v_j))$; 2. $vdir(v_i) \neq (0,0,0)$ and $v_i$ and $v_j$ are connected in a non-dominant direction of $v_i$; 3. $v_i$ and $v_j$ are not a part of a $G(1,2,0)$ subgraph (Section 4.3).

**edge** is a link between vertices

**erosion** Let $A$ be the set of raster cells, e.g. containing point cloud points, which is modified with a set $B$. $B_x$ denotes the translation of $B$ in direction of vector $x$. Then, $A \oplus B = \{x | B_x \cap A \neq \emptyset\}$. See Section 3.2.1.

**grid graph** a three-dimensional grid graph is an $m \times n \times r$ graph that is the graph Cartesian product of path graphs on $m$, $n$ and $r$ vertices. The grid graph is denoted as $G(m,n,r)$ (Section 4.2.2).

**G(m,n,r)** denotes the type of a grid graph (Section 4.2.2).

**genus** of a watertight surface is a number representing the maximum number of cuttings along non-intersecting closed simple curves without that the resultant manifold gets disconnected. Not strictly correct, but intuitive, one can think of the number of loops formed by an object.

**graph** is a structure consisting of vertices and edges. It is denoted as $G(V,E)$, where $V$ are the vertices and $E$ are the edges.

**HARPER** abrivates **H**istogram **A**nalysis & **R**etraction of **P**oints **E**stimated **R**adii. It is a new method to estimate diameters in point clouds.

**infinum** is the greatest lower bound of a set $S$, defined as a quantity $m$ such that no member of the set is less than $m$.

**level set** is the set of values $x$ for which a real-valued function $f(x)$ is equal to a given constant (Section 3.3.2).

**medial axis** of an object is the set of points having more than one closest point on the object boundary (Section 3.2).

**merging** The union of two vertices. (Section 4.3).

**octree** an octree is a hierarchical subdivision of a starting cube into 8 equally sized subcubes, which are the octree cells. The octree space is modeled as a cubical region consisting of $2^n \times 2^n \times 2^n$ unit cubes, where $n$ is the subdivision depth. Each unit cube has value 0 or 1, depending on whether it contains data points or not (Section 4.2.2).

**octree cell** see octree

**octree-graph** is a graph extracted from and octree subdivision, representing the adjacency of the octree cells. Let $OC_i, i = 1..n$ be a collection of octree cells. And let $CS_{jk}$, for some $j \neq k, ; j,k \leq n$ be the adjacent sides of the octree cells. The octree-graph $OG(V,E)$ contains the vertices $V$ dual to $OC_i$ connected via the edges $E$ dual to $CS_{jk}$ (Section 4.2.2).

**octree space** see octree

**SkelTre** is the abbrevation for **Skel**etonization of **Tre**es (Chapter 4).

**tree delineation** Tree delineation is the process to automatically derive individual trees from a point cloud (Section 6.2.4).

**supremum** is the least upper bound of a set $S$, defined as a quantity $M$ such that no member of the set exceeds $M$.

**V-Pair** is generated by E-Pairs and is a valid pair of vertices to be merged by the SkelTre algorithm. Two vertices $v_i$ and $v_j$ both incident to a vertex $v_c$ are called a V-Pair if: 1 the labels of edges $v_i v_c$ and $v_j v_c$ are identical; 2. $vdim(v_i \oplus v_j) \leq \max(vdim(v_i), vdim(v_j))$ (Section 4.3).

**vertex** element of a graph that links via edges to other vertices.

**vertex dimension** $vdim(v_i)$ is the number of distinct edge labels associated to a vertex $v_i$ (Section 4.3).

**vertex direction** is the direction of a vertex $v_i$ is denoted as $vdir(v_i)$. The sum $vdir(v_i)$ over the distinct associated edge labels of a vertex $v_i$ is called the vertex direction (Section 4.3).

**vertex norm** is the sum over the absolute values of the vertex direction. Let $vdir(v_i) = (x_1, x_2, x_3)$. The norm of $v_i$ is $norm(vdir(v_i)) = norm(x_1, x_2, x_3) = |x_1|+|x_2|+|x_3|$ (Section 4.3).

**voronoi diagram** is a decomposition of a metric space determined by distances. The Definition can be found in Section 3.2.3

# Bibliography

[Adams, 1979] Adams, D. (1979). *The Hitchhiker's Guide to the Galaxy*. Pan Books, United Kingdom.

[Aim@Shape, 2008] Aim@Shape (2008). Aim@Shape repository. http://shapes.aim-at-shape.net/.

[Amenta et al., 2001] Amenta, N., Choi, S., and Kolluri, R. K. (2001). The power crust, unions of balls and the medial axis transform. *Computational Geometry: Theory and Applications*, 19(2-3):127–153.

[Andersen et al., 2006] Andersen, H. E., Reutebuch, S. E., and McGaughey, R. J. (2006). A rigorous assessment of tree height measurements obtained using airborne LiDAR and conventional field methods. *Canadian Journal of Remote Sensing*, 32(5):355–366.

[Arthur and Vassilvitskii, 2005] Arthur, D. and Vassilvitskii, S. (2005). On the worst case complexity of the k-means method. *Technical Report, Stanford*, 698.

[Attene et al., 2003] Attene, M., Biasotti, S., and Spagnuolo, M. (2003). Shape understanding by contour-driven retiling. *The Visual Computer*, 19:127–138.

[Baker et al., 1992] Baker, V., Carr, M., Gulick, V., Williams, C., and Marley, M. (1992). Channels and valley networks. *Mars*, 1:493–522.

[Bartelink, 1997] Bartelink, H. (1997). Allometric relationships for biomass and leaf area of beech (fagus sylvatica l). *Annales de Science Forestiere*, 57:39–50.

[Biasotti et al., 2008] Biasotti, S., Attali, D., Boissonnat, J.-D., Edelsbrunner, H., Elber, G., Mortara, M., di Baja, G. S., Spagnuolo, M., Tanase, M., and Veltkamp, R. (2008). Skeletal structures. *In: Shape Analysis and Structuring,Leila De Floriani, Michaela Spagnuolo (Editors), Chapter 6*.

[Blais, 2004] Blais, F. (2004). Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231–240.

[Blum, 2007] Blum, H. (2007). A transformation for extracting new descriptors of shape. *In Proceedings: Models for perception of speech and visual form*, pages 362–380.

[Bucksch and Fleck, 2009] Bucksch, A. and Fleck, S. (2009). Automated detection of branch dimensions in woody skeletons of leafless fruit tree canopies. *In Proceedings: SilviLaser 2009*.

[Bucksch and Fleck, 2010] Bucksch, A. and Fleck, S. (2010). Automated detection of branch dimensions in woody skeletons of fruit tree canopies. *Photogrametric Engeneering and Remote Sensing*, Special Issue of SilviLaser 2009:in print.

[Bucksch and Lindenbergh, 2008] Bucksch, A. and Lindenbergh, R. (2008). Campino - a skeletonisation method for point cloud processing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63:115–127.

[Bucksch et al., 2009a] Bucksch, A., Lindenbergh, R., and Menenti, M. (2009a). Skeltre - fast skeletonization of imperfect point clouds of botanic trees. *In Proceedings: Eurographics/ACM Siggraph 3D Object Retrieval Workshop*, pages 13–20.

[Bucksch et al., 2010] Bucksch, A., Lindenbergh, R., and Menenti, M. (2010). Skeltre - robust skeleton extraction from imperfect point clouds. *The Visual Computer*, 26(10):1283–1300.

[Bucksch et al., 2009b] Bucksch, A., Lindenbergh, R., Menenti, M., and Rahman, M. Z. (2009b). Skeleton-based botanic tree diameter estimation from dense LiDAR data. *Proceedings of SPIE Vol. 7460 (SPIE, Bellingham, WA 2009)*.

[Buddenbaum and Seeling, 2007] Buddenbaum, H. and Seeling, S. (2007). Derivation of tree height and crown closure from airborne LiDAR imagery. *Geophysical Research Abstracts*, 9:322–336.

[Burger, 1945] Burger, H. (1945). Holz, Blattmenge und Zuwachs - Die Buche. *Mitteilung der Schweizer Anstalt Forstlichen Versuchswesens*, 26:419–468.

[Chen and Huang, 1988] Chen, H. H. and Huang, T. S. (1988). A survey of construction and manipulation of octrees. *Computer Vision, Graphics, and Image Processing*, 43(3):409–431.

[Clark et al., 2004] Clark, M. L., Clark, D. B., and Roberts, D. A. (2004). Small-footprint LiDAR estimation of sub-canopy elevation and tree height in a tropical rain forest landscape. *Remote Sensing of Environment*, 91(1):68–89.

[Cline et al., 2009] Cline, D., Jeschke, S., White, K., Razdan, A., and Wonka, P. (2009). Dart throwing on surfaces). *Computer Graphics Forum,*, 28:1217–1226.

[Cole-McLaughlin et al., 2004] Cole-McLaughlin, K., Edelsbrunner, H., Harer, J., Natarajan, V., and Pascucci, V. (2004). Loops in Reeb graphs of 2-manifolds. *Discrete Computational Geometry*, 32(2):231–244.

[Cornea and Min, 2007] Cornea, N. D. and Min, P. (2007). Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548.

[Cornea et al., 2005] Cornea, N. D., Silver, D., and Min, P. (2005). Curve-skeleton applications. In *IEEE Visualization Conference*, pages 95–102.

[Côtè et al., 2009] Côtè, J.-F., Widlowski, J.-L., Fournier, R. A., and Verstraete, M. M. (2009). The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial LiDAR. *Remote Sensing of Environment*, 113(5):1067–1081.

[Dai et al., 2009] Dai, M., Zhang, X., Zhang, Y.-K., and Jaeger, M. (2009). Segmentation of point cloud scanned from trees. *In Proceedings: Workshop on 3D content and applications with ACCV 2009*, Xi'an, China.

[de Berg et al., 2008] de Berg, M., Cheong, O., van Kreveld, M., and Overmars, M. (2008). *Computational Geometry - Algorithms and Applications (Third Edition)*. Springer Verlag Inc.

[Dey, 2007] Dey, T. K. (2007). *Curve and Surface Reconstruction*. Cambridge University Press.

[Dey and Sun, 2006] Dey, T. K. and Sun, J. (2006). Defining and computing curve-skeletons with medial geodesic function. *In Proceedings: 4th Eurographics symposium on Geometry processing*, pages 143–152.

[Fleck, 2002] Fleck, S. (2002). *Integrated analysis of relationships between 3D-structure, leaf photosynthesis and branch transpiration of Fagus sylvatica and Quercus petraea in a mixed forest*. Dissertation at University of Bayreuth, Bayreuth Institute for Terrestrial Ecosystem Research.

[Fleck et al., 2010] Fleck, S., Embree, C. G., and Nichols, D. (2010). The systematic influence of crop load, spur type, 3D canopy structure, and leaf zonal chlorosis on leaf photosynthesis of honeycrisp apple trees. *Acta Horticulturae (accepted)*.

[Fleck et al., 2007] Fleck, S., Obertreiber, N., Schmidt, I., Brauns, M., Jungkunst, H., and Leuschner, C. (2007). Terrestrial LiDAR measurements for analysing canopy structure in an old-growth forest. *International Archives of the Photogrammetry*, 26(3/W52):125–129.

[Fleck et al., 2004] Fleck, S., van der Zande, D., Schmidt, M., and Coppin, P. (2004). Reconstructions of tree structure from laser-scans and their use to predict physiological properties and processes in canopies. *International Archives of the Photogrametry, Remote Sensing and Spatial Information Sciences*, 36(8/W2):119–123.

[Fugro Aerial Mapping, 2008] Fugro Aerial Mapping (2008). Fli-map400. http://www.flimap.nl/.

[Gorte, 2006] Gorte, B. (2006). Skeletonization of laser-scanned trees in the 3d raster domain. *In Proceedings: 3DGeoInfo06, Malaysia*.

[Gorte and Pfeifer, 2004] Gorte, B. and Pfeifer, N. (2004). Structuring Laser-Scanned Trees Using 3D Mathematical Morphology. *IAPRS*, XXXV(B5):929–933.

[Henning and Radtke, 2006] Henning, J. G. and Radtke, P. (2006). Detailed stem measurements of standing trees from ground-based scanning LiDAR. *Forest Science*, 52:67–80.

[Henning and Radtke, 2008] Henning, J. G. and Radtke, P. (2008). Multiview range-image registration for forested scenes using explicitly-matched tie points estimated from natural surfaces. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(1):68–83.

[Hodge, 2010] Hodge, R. A. (2010). Using simulated terrestrial laser scanning to analyse errors in high-resolution scan data of irregular surfaces. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(2):227 – 240.

[Hyyppä et al., 2005] Hyyppä, J., Mielonen, T., Hyyppä, H., Maltamo, M., Yu, X., Honkavaara, E., and Kaartinen, H. (2005). Using individual tree crown approach for forest volume extraction with aerial images and laser point clouds. *In Proceedings: ISPRS Workshop Laser scanning 2005*, XXXV(B5):929–933.

[Jonckheere et al., 2004] Jonckheere, I., Fleck, S., Nackaerts, K., Muys, B., P, C., Weiss, M., and Baret, F. (2004). A review of methods for leaf area index estimation of forest stands focused on digital hemispherical photography. *Agricultural and Forest Meteorology*, 121(1):19–35.

[Kalliovirta and Tokola, 2005] Kalliovirta, J. and Tokola, T. (2005). Functions for estimating stem diameter and tree age using tree height, crown width and existing stand database information. *Silva Fennica*, 39(2):227248.

[Klette et al., 1998] Klette, R., Rosenfeld, A., and Sloboda, F. (1998). *Advances in Digital and Computational Geometry*. Springer Verlag Inc.

[Knuth, 1998] Knuth, D. E. (1998). *The Art of Computer Programming*. Addison-Wesley, volume 3, 2nd edition.

[Korpela et al., 2007] Korpela, I., Dahlin, B., Schfer, H., Bruun, E., Haapaniemi, F., Honkasalo, J., Ilvesniemi, S., Kuutti, V., Linkosalmi, M., Mustonen, J., Salo, M., Suomi, O., and Virtanen, H. (2007). Single-tree forest inventory using LiDAR and aerial images for 3D treetop positioning, species recognition, height and crown width estimation. *International Archieves of Photogrammetry and Remote Sensing and spatial information sciences*, XXXVI(3):227–233.

[Köstner et al., 2004] Köstner, B., Schmidt, M., Falge, E., Fleck, S., and Tenhunen, J. D. (2004). Atmospheric and structural controls on carbon and water relations in mixed-forest stands. *In Book: Matzner, E., Temperate Forest Ecosystems response to Changing Environment: Watershed Studies in Germany*, Ecological Studies 172:69–98.

[Laasasenaho, 1982] Laasasenaho, J. (1982). *Taper curve and volume functions for pine, spruce and birch*. Number 108 in Communicationes Instituti Forestalis Fenniae. Finnish Forest Research Institute.

[Langetepe and Zachmann, 2006] Langetepe, E. and Zachmann, G. (2006). *Geometric data structures for computer graphics*. A.K. Peters, Ltd., Wellesley.

[Leica Geosystems, 2009] Leica Geosystems (2009). Software: Cyclone 6.0. *3D Point Cloud Processing Software*.

[Löffler and van Kreveld, 2007] Löffler, M. and van Kreveld, M. (2007). Largest bounding box, smallest diameter, and related problems on imprecise points. *Department of Information and Computing Sciences, Utrecht University Technical Report*, UU-CS-2007-025:1–18.

[Lucchese and Mitra, 2001] Lucchese, L. and Mitra, S. (2001). Colour image segmentation: a state-of-the-art survey. *Proceedings - Indian National Science Academy part A*, 67(2):207–222.

[Meia and Durrieu, 2004] Meia, C. and Durrieu, S. (2004). Tree crown delineation from digital elevation models and high resolution imagery. In *International Archives of Photogrammetry and Remote Sensing*, volume XXXVI, pages 218–223.

[Meyers et al., 1992] Meyers, D., Skinner, S., and Sloan, K. (1992). Surfaces from contours. *ACM Transactions on Graphics*, 11(3):228–258.

[Milnor, 1963] Milnor, J. (1963). *Morse Theory*. Princeton University Press.

[Musser and Saini, 1997] Musser, D. and Saini, A. (1997). *Stl Tutorial and Reference Guide: C++ Programming With the Standard Template Library*. Addison-Wesley.

[Naesset and Okland, 2002] Naesset, E. and Okland, T. (2002). Estimating tree height and tree crown properties using airborne scanning laser in a boreal nature reserve. *Remote Sensing of Environment*, 79(1):105–115.

[Niklas, 1994] Niklas, K. (1994). *Plant allometry: the scaling of form and process*. The University of Chicago Press.

[O'Rourke, 2005] O'Rourke, J. (2005). *Computational geometry in C, 2nd Edition*. Cambridge University Press.

[Palagyi et al., 2001] Palagyi, K., Sorantin, E., Balogh, E., Kuba, A., Halmai, C., Erdohelyi, B., and Hausegger, K. (2001). A sequential 3D thinning algorithm and its medical applications. *In Proceedings: IPMI '01*, pages 409–415.

[Pascucci et al., 2007] Pascucci, V., Scorzelli, G., Bremer, P.-T., and Mascarenhas, A. (2007). Robust on-line computation of reeb graphs: simplicity and speed. *ACM Transactions on Graphics*, 26(3):58.

[Pemmaraju and Skiena, 2003] Pemmaraju, S. and Skiena, S. (2003). *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Cambridge University Press.

[Pfeifer and Briese, 2007] Pfeifer, N. and Briese, C. (2007). Laser scanning principles and applications. *In Proceedings: Geo-Sibir, III International Scientific Conference Novosibirsk,*, pages 93 – 112.

[Pfeifer et al., 2004] Pfeifer, N., Gorte, B., and Winterhalder, D. (2004). Automatic reconstruction of single trees from terrestrial laser scanner data. *In Proceedings of 20th ISPRS Congress*, pages 114–119.

[Popescu et al., 2003] Popescu, S. C., Wynne, R. H., and Nelson, R. F. (2003). Measuring individual tree crown diameter with LiDAR and assessing its influence on estimating forest volume and biomass. *Canadian Journal of Remote Sensing*, 29(5):564–577.

[Rahman and Gorte, 2008] Rahman, M. and Gorte, B. (2008). Individual tree detection based on desities of high points from high resolution airborne data. *In Proceedings: GEOBIA 2008*.

[Rahman and Gorte, 2009] Rahman, M. Z. and Gorte, B. (14.-16. October 2009). A new method for individual tree measurement from airborne LiDAR. *In Proceedings: SilviLaser 2009, Colledge Station, Texas*.

[Rahman et al., 2009] Rahman, M. Z., Gorte, B., and Bucksch, A. (1.-2. September 2009). A new method for individual tree delineation and undergrowth removal from high resolution airborne LiDAR. *In Proceedings: ISPRS workshop Laserscanning. Paris, France*.

[Rahman, 2011] Rahman, M. Z. A. (2011). Estimation of composite hydrological roughness overland with high density airborne laser scanning. *PhD Thesis, TU Delft*.

[Reeb, 1946] Reeb, G. (1946). Sur les points singuliers d'une forme de Pfaff completement integrable ou d'une fonction numérique. *Comptes Rendus Acad. Sciences*, 222:847–849.

[Richardson, 2003] Richardson, I. (2003). *H. 264 and MPEG-4 video compression*. Wiley Online Library.

[Rijkswaterstaat, 2008] Rijkswaterstaat (2008). Actual height model of the netherlands. http://www.ahn.nl/.

[Roloff, 1986] Roloff, A. (1986). Morphologie der Kronenentwicklung von Fagus sylvatica L. (Rotbuche) unter besonderer Berücksichtigung möglicherweise neuartiger Veränderungen. *Ph.D. Thesis at the University of Göttingen*.

[Serra, 1982] Serra, J. (1982). *Image analysis and mathematical morphology*. Academic Press, London.

[Shan and Todd, 2008] Shan, J. and Todd, C. (2008). *Topographic Laser Ranging and Scanning (Eds.)*. CRC Press.

[Shinagawa et al., 1991] Shinagawa, Y., Kunii, T. L., and Kergosien, Y. (1991). Surface coding based on morse theory. *IEEE Computer Graphics and Applications*, 11:66–78.

[Straatsma and Baptist, 2008] Straatsma, M. W. and Baptist, M. J. (2008). Floodplain roughness parameterization using airborne laser scanning and spectral remote sensing. *Remote Sensing of Environment*, 112(3):1062–1080.

[Strahler et al., 2008] Strahler, A., Jupp, D., Woodcock, C., Schaaf, C., Yao, T., Zhao, F., Yang, X., Lovell, J., Culvenor, D., Newham, G., Ni-Miester, W., and Boykin-Morris, W. (2008). Retrieval of forest structural parameters using a ground-based LiDAR instrument(echidna). *Canadian Journal of Remote Sensing*, 34(2):426–440.

[Tomokaki, 2005] Tomokaki, T. (2005). Predicting individual stem volumes of sugi (cryptomeria japonica d. don) plantations in mountainous areas using small-footprint airborne LiDAR. *Journal of Forest Research*, 10(4):305–312.

[Verroust and Lazarus, 2000] Verroust, A. and Lazarus, F. (2000). Extracting skeletal curves from 3D scattered data. *The Visual Computer*, 16:15–25.

[Wilson, 1985] Wilson, R. J. (1985). *Introduction to Graph theory third edition*. Longman Scientific & Technical, Essex.

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

[Yan et al., 2009] 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 Proceedings: 11th IEEE International conference on Computer-Aided Design and Computer Graphics*.

[Zhou et al., 1998] Zhou, Y., Kaufman, A., and W.Toga, A. (1998). Three-dimensional skeleton and centerline generation based on an approximate minimum distance field. *The Visual Computer*, 14(7):303–314.

[Zoller und Fröhlich, 2009] Zoller und Fröhlich (2009). Software: Z+f laser control. *3D Point Cloud Processing Software*.

[Zomorodian, 2005] Zomorodian, A. J. (2005). *Topology for computing*. Cambridge University Press.

# Publications

## Publications contributing to this thesis

### Journal Publications

2010: *SkelTre - Robust skeleton extraction from imperfect point clouds* Alexander Bucksch, Roderik Lindenbergh and Massimo Menenti The Visual Computer, Vol.26, No. 10, pp. 1283-1300, DOI: 10.1007/s00371-010-0520-4

2010: *Automated detection of branch dimensions in woody skeletons of fruit tree canopies* Alexander Bucksch, Stefan Fleck Photogrammetric Engineering and Remote Sensing (in print)

2008: *CAMPINO - A skeletonization method for point cloud processing* Alexander Bucksch and Roderik Lindenbergh ISPRS Journal of Photogrammetry and Remote Sensing, Vol.63, No.1, pp. 115-127

### Peer Reviewed Conference Publications

2009: *Skeltre - Fast skeletonization of imperfect point clouds of botanic trees* Alexander Bucksch, Roderik Lindenbergh and Massimo Menenti 3D Object Retrieval Workshop/ Eurographics 2009, Munchen 28.March-3.April 2009

2009: *Automated detection of branch dimensions in woody skeletons of leafless fruit tree canopies* Alexander Bucksch and Stefan Fleck In Proceedings SilviLaser 2009 , 14.-16. October 2009 Austin,Texas

2009: *A new method for individual tree delineation and undergrowth removal from high resolution airborne LiDAR* M.Z.A. Rahman, B.G.H. Gorte, A.K. Bucksch ISPRS Workshop on Laser Scanning 2009, 1.-2. September 2009 Paris, France

### Abstract Reviewed Conference Publications

2009: *Skeleton-based botanic tree diameter estimation from dense LiDAR data* Alexander Bucksch, Roderik Lindenbergh, Massimo Menenti, Muhammad Z. Raman at Opticts and Photonics 2009, San Diego(CA). In Lidar Remote Sensing for Environmental Monitoring X, edited by Upendra N. Singh, Proceedings of SPIE Vol. 7460 (SPIE, Bellingham, WA 2009) 746007.

2006: *Skeletonization and segmentation of point clouds using octrees and graph theory* Alexander Bucksch and Heinz Appel van Wageningen ISPRS Symposium: Image Engineering and Vision Metrology Vol. XXXVI. ISPRS Commission V Symposium (pp. 1-6). Dresden: Dresden University of Technology.

# Publications not related to the thesis

## Journal Publications

2008: *3D Laser imaging as a valuable tool to specify changes in breast shape after augmentation mammaplasty* Danielle L. Esme, Alexander Bucksch, Werner H. Beekman Aesthetic Plastic Surgery Volume 33, Number 2, page 191-195

## Peer Reviewed Conference Proceedings

2009: *A new method for individual tree measurement from airborne LiDAR* Muhamad Z.Abd Rahman, Ben Gorte, Alexander Bucksch In Proceedings: SilviLaser 2009 , 14.-16. October 2009 Austin,Texas

## Invited Talks

2007: *Error budget of terrestrial laserscanning: Influence of the intensity remission of the scan quality* Alexander Bucksch, Roderik Lindenbergh and Jane van Ree Proceedings of the III International Scientific Congress Geo-Siberia 2007, 23-27 April, Novosibirsk, Vol. I, 2nd part,, Geodesy, Geoinformatics, Cartography, Markscheider, pp. 113-122, ISBN 978-5-87693-229-7 and ISBN 978-5-87693-231-0

## Abstract Reviewed Conference Publications

2010: Massimo Menenti, Muhamad Z. Abd Rahman, Alexander Bucksch, Roderik Lindenbergh and Hieu van Duong. *Retrieval of vegetation and surface properties with terrestrial, airborne and space-borne laser scanners.* In Proceedings: 3rd International Symposium in Recent advances in quantitative remote sensing, Torrent, Spain. (in preperation)

and also presented at: In Proceedings: Symposium Climate, Management and topography impacts on vegetation: A tribute to Dr. Jerry Ritchie, Long Beach, California, USA.

2010: *Woody biovolume extraction from laser scanned trees* Alexander Bucksch, Stefan Fleck, Sabiene Rumpf,Peter Rademacher In Proceedings: Silvilaser 2010, 14-17 September 2010, Freiburg Germany

2009: *Structural monitoring of Tunnels using terrestrial laser scanning* Roderik Lindenbergh, Lukasz Uchanski, Alexander Bucksch, Rinske van Gosliga Reports of Geodesy, Special Issue of the IX Konferencji naukowo-technicznej ”Aktualne Problemy w Geodezji Inz.ynieryjnej, 27./28. March 2009 in Warsawa

2009: *Applications for point cloud skeletonization in forestry and agriculture* Alexander Bucksch, Roderik Lindenbergh Reports of Geodesy, Special Issue of the IX Konferencji naukowo-technicznej ”Aktualne Problemy w Geodezji Inz.ynieryjnej, 27./28. March 2009 in Warsawa

2007: *Error budget of terrestrial laser scanning: Influence of the incidence angle on the scan quality* Sylvie Soudarissanane, Jane van Ree, Alexander Bucksch and Roderik Lindenbergh Proceedings 3D-NordOst, 10. Anwendungsbezogener Workshop zur Erfassung, Modellierung, Verarbeitung und Auswertung von 3D-Daten, Berlin, 06.-07. December 2007

2007: *3D Buildings modeling Based on a combination of techniques and methodologies* GeorgetaPop, Alexander Bucksch, Ben Gorte XXI CIPA International Symposium, 1-6 October 2007,Athens, Greece p. 1-5

2007: *Combining modern techniques for urban 3D modeling* Georgeta Pop, Alexander Bucksch IEEE International Geoscience and Remote Sensing Symposium pp. 1-4, 23-27 July 2007, Barcelona, Spain.

2004: *A system for recording 3D information with applications in the measurement of plant structure* Thomas Mangoldt , Winfried Kurth , Alexander Bucksch , Peter Wernecke , Wulf Diepenbrock Poster on the ”4th International Workshop on Functional-Structural Plant Models” (FSPM04), Montpellier (France), 7.-11. 6. 2004.

## Other Publications

2008: *Technical Report - FLI-MAP data possibilities for forest inventory* Alexander Bucksch, Ben Gorte, Muhammad Z. Abd Rahman ICT Dienst Rijkswaterstaat

2006: *3D model generation with laser scanners - Approaches towards the improvement of CFD input data* Alexander Bucksch, Adamantios Kagkaras Leonardo Times, June 2006

2005: *Development of an integrated algorithm for surface reconstruction and point reduction in point clouds* (in german) Alexander Bucksch Master Thesis at BTU Cottbus under supervision of Prof. Winfried Kurth, February 2005

# Index

Actual Height model of The Netherlands, 83
algorithm classes, 37
algorithmic requirements, 14
allometric relationships, 69
attributes, 15

Big-O notation, 20
boundary, 13
branch parameters, 69
branching, 17
branching regions, 17

CAMPINO, 34
Centeredness, 58
classes, 14
computational complexity, 20, 53
computational efficiency, 14, 53
computational framework, 37
contour, 31
convergence, 58

descriptive requirements, 13
diameter at breast height, 82
dilatation, 26
directed graphs, 47
distance transform, 27

E-Pair, 38, 43
elongation direction, 17
erosion, 26
error characteristics, 53

FLI-MAP 400, 82

genus, 46
graph, 14
graph embedding, 38
graph reduction, 33
grid graph, 41

handling of imperfect data, 14
HARPER, 69
histogram of point distances to the skeleton, 62
hit-and-miss-transform, 26

imperfect point cloud, 12
implementation, 53

labeling, 38
line segment, 14

median bin, 75
metric representation, 13, 53
morphological thinning, 26

non-branching, 17
non-branching regions, 17
norm of a vertex, 43

occlusion effects, 19
octree, 38
octree cell, 37
octree space, 38
octree-graph, 38, 39
orchard trees, 69
outer form, 12

peak bin, 86
point cloud, 2
poles, 29
post processing, 58
power crust, 29
power diagram, 29

reduction, 9
Reeb-graph, 31
reference bin, 75
reliability, 38, 53
requirements, 15
robustness criterion, 38
running times, 65

skeleton, 6
skeleton extraction requirements, 14
skeleton graph, 14
skeleton-graph, 38
SkelTre, 37, 53
SkelTre-skeleton, 9
smallest diameter of an imprecise point set, 85
space, 13
structuring element, 26

terrestrial laser scan data, 58

topological preservation, 13, 53
topological space, 13
tree-delineation, 69
trellis system, 71

V-Pair, 38, 43
vertex dimension, 9, 42
vertex direction, 43
Voronoi Diagram, 29

# Curriculum Vitae

Alexander Bucksch was born in 1976 in Böblingen, Germany. He visited the Carlo-Schmid Gymnasium in Tübingen in to get his university entrance diploma in 1995. After that he went for alternative military service, before he started to study Mediatechnology at the TU Ilmenau, Germany. He graduated in Information-and Mediatechnology in 2004 at the BTU Cottbus, Germany. After working for several month on Voronoi-based image compression at the Institute for semiconductor physics in Frankfurt/Oder, Germany, he started to participate in the canopy analyzer project. This project was a cooperation between the Delft University of Technology, the University of Göttingen, Germany and the terrestrial laser scanner manufacturer Zoller und Fröhlich. The canopy analyzer project led to the PhD thesis you just read.