FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

# Reeb Graph Based Image Representation for Phenotyping of Plants

DIPLOMARBEIT

zur Erlangung des akademischen Grades

**Diplom-Ingenieurin**

im Rahmen des Studiums

**Visual Computing**

eingereicht von

**Ines Janusch**
Matrikelnummer 0525109

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: o.Univ.-Prof. Dipl.-Ing. Dr. Walter G. Kropatsch

Wien, 10.04.2014

(Unterschrift Verfasserin)

(Unterschrift Betreuung)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

# Reeb Graph Based Image Representation for Phenotyping of Plants

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

**Diplom-Ingenieurin**

in

**Visual Computing**

by

**Ines Janusch**
Registration Number 0525109

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: o.Univ.-Prof. Dipl.-Ing. Dr. Walter G. Kropatsch

Vienna, 10.04.2014

(Signature of Author) (Signature of Advisor)

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.ac.at

# Erklärung zur Verfassung der Arbeit

Ines Janusch
Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

____________________________ (Ort, Datum)

____________________________ (Unterschrift Verfasserin)

# Acknowledgements

I would like to thank my supervisor Walter Kropatsch for his guidance during my work on this thesis; for the constant opportunity of discussion and feedback and for his encouragement to gain extra experience by participating in workshops.

My thanks go to the members of PRIP for their input and feedback in presentations and discussions. To everyone who helped with proof-reading and advise (as always, on a tight schedule): thank you for your help.

I would like to thank my family and friends who accompanied and supported me, not only through this period of writing my thesis, but throughout the duration of my studies.

And finally, I would like to thank Clemens, who by now is one, of probably only a few, civil engineers with an extensive knowledge of Reeb graphs. Thank you for feigning interest so well and for putting up with my need for discussions and opinions.

# Abstract

While genotypes are defined as the set of genes an organism holds, its phenotype is defined as the set of its observable characteristics. To determine the correlation of genotype and phenotype or how a phenotype is affected by environmental conditions, an evaluation on large datasets is needed. An automatic analysis of image data and extraction of characteristics allows for large-scale evaluations.

This thesis presents a comparison of two types of graph-based image representations: medial axis transformation and Reeb graphs and evaluates the feasibility of using these representations in image based plant phenotyping.

A presegmented binary image of roots (of the plant *Arabidopsis thaliana*) is the basis for generating the well-known medial axis and the Reeb graphs. For phenotyping of plants their root structure is analysed. The main characteristics used here are branching points, branch endings as well as the length and width of individual branches. These characteristics are captured by the presented graph representations. For the computation of the Reeb graphs two different Morse functions are used: height function and geodesic distance.

As the roots are pictured as 2D images, the projection of a 3D structure to a 2D space might result in an overlap of branches in the image. One major advantage, when analysing roots based on Reeb graphs, is posed by the ability to immediately distinguish between branching points and overlaps in the root structure as an overlap introduces a cycle and thereby a certain type of node (saddle - merge) in the Reeb graph. This differentiation is not as easily possible by a medial axis representation or by an analysis solely based on contours.

In order to use the advantages of different representations and the characteristics provided by them, a possibility to combine different graph representations of one root image is needed. Therefore the equality of graphs is evaluated. This thesis shows that all three representations of a root are either isomorphic graphs or have a common subgraph.

A new normalised graph representation for root images is introduced. This representation is based on a (Reeb) graph and represents the branching structure of a root as side branches to the left and right of a main root. This normalised representation focuses on the branching pattern and the length of branches and allows for an efficient comparison and evaluation of root images.

The graph representations computed for this thesis were compared to human generated ground truth. Here the Reeb graphs based representations approximate the ground truth best.

# Kurzfassung

Ähnlich zu den als Menge der Gene eines Organismus definierten Genotypen, sind Phänotypen definiert als die Menge der beobachtbaren Eigenschaften dieses Organismus. Um den Zusammenhang von Genotypen und Phänotypen sowie den Einfluss von Umweltbedingungen auf den Phänotyp zu bestimmen, werden große Datensätze evaluiert. Eine automatische Analyse von Bilddaten und Extraktion von Charakteristiken ermöglicht die Auswertung dieser großangelegten Studien.

Diese Diplomarbeit widmet sich dem Vergleich zweier graphbasierter Bild-Repräsentation: Medial Axis Transformation und Reeb Graphen und analysiert die Einsatzmöglichkeit dieser Abbildungen in der bildbasierten Pflanzen Phänotypisierung.

Ein segmentiertes Binärbild der Wurzeln (der Pflanze *Arabidopsis thaliana*) bildet die Grundlage zur Berechnung der bekannten Medial Axis und der Reeb Graphen. Zur Phänotypisierung von Pflanzen wird ihre Struktur analysiert. Die wichtigsten Charakteristiken dabei sind: Verzweigungen, Enden von Zweigen, sowie die Länge bzw. Dicke der einzelnen Zweige. Diese Eigenschaften werden von den verwendeten Graph-Repräsentationen erfasst. Zur Berechnung der beiden Reeb Graphen werden zwei unterschiedliche Morse Funktionen verwendet: die Höhenfunktion sowie die geodätische Distanz.

Da die Pflanzen als 2D Datensatz abgebildet werden, findet eine Projektion der 3D Wurzelstruktur in den 2D Raum des Bildes statt. Diese Projektion kann in Überlappungen einzelner Wurzelzweige im Bild enden. Ein bedeutender Vorteil der Reeb Graph Repräsentationen ist durch die Möglichkeit, Verzweigungspunkte sofort von Überlappungen unterscheiden zu können, gegeben. Dies beruht auf der Eigenschaft der Reeb Graphen, dass durch die Überlappungen der Wurzeln im Bild Zyklen in der Graphstruktur entstehen. Diese Zyklen erzeugen einen Knoten vom Typ Sattel (merge) im Graph. Eine Unterscheidung zwischen Überlappungen und Verzweigungen ist, für eine Darstellung basierend auf der Medial Axis oder einer Analyse der Wurzelstruktur anhand der Kontur, nicht so leicht möglich.

Um die Vorteile verschiedener Darstellungen und die entsprechenden Charakteristiken zu verwenden, wird eine Kombination der unterschiedlichen graphbasierten Repräsentationen benötigt. Dazu muss die Gleichwertigkeit der Graphen bewertet werden. Im Rahmen dieser Diplomarbeit zeigt sich, dass die Graphen aller drei Darstellungen entweder isomorph sind oder einen gemeinsamen Teilgraphen enthalten.

Letztendlich wird eine neue normalisierte graphbasierte Darstellung für Wurzelbilder präsentiert. Diese Repräsentation basiert auf (Reeb) Graphen und stellt die Verzweigungsstruktur einer Wurzel als Seitenäste zur linken und rechten Seite der Hauptwurzel dar. Diese normaliserte Dar-

stellung konzentriert sich auf die Abbildug der Verzweiungsstruktur und der Länge der Wurzeln und erlaubt effizienten Vergleich und Evaluierung von Wurzelbildern.
Die vorgestellten graphbasierten Repräsentationen wurden mit menschlich generierten Referenzdaten verglichen. Die Reeb Graph Repräsentationen approximieren diese Referenzwerte am besten.

# Contents

# CHAPTER 1

# Introduction

This thesis addresses the problem of shape analysis and representation within the context of image based plant phenotyping.
A first definition of phenotypes was given by Johannsen in 1911 [26]:

> "All "types" of organisms, distinguishable by direct inspection or only by finer methods of measuring or description, may be characterised as "phenotypes"."

While an organism's genotype is defined as the set of genes this organism holds, its phenotype is defined by its observable characteristics (from Greek: *phainein* = to show).
Gregor Mendel first introduced the concept of dominance, which stated that some inherited traits were dominant over others as a heterozygot (genotype consisting of two different alleles) offspring could show the same phenotype as its homozygot (genotype consisting of two identical alleles) parent. However, the relationship between genotypes and phenotypes proved to be more complex than this simple dominant pattern described by Mendel [34].
While the phenotype of an organism is based on its genes, it is just as well influenced by environmental conditions. While the genome of certain plants or animals (so called model organisms) is completely characterised [40], their phenome (the full set of phenotypes of an individual) can never be completely characterised as phenotypes vary strongly [23].
Apart from this application in molecular biology, this thesis is set within the field of image processing, more precisely within the area of topological image analysis and shape representation. The following keywords describe the overall context:

- Graph Theory
- Topological Image Analysis
- Shape Extraction, Representation and Comparison

## 1.1 Problem Statement

To address this major question in biology, how genotypes translate into phenotypes, the ability to phenotype a large number of individuals is needed. Therefore an efficient extraction and representation of characteristics is needed.
For phenotyping of plants their root development, the architecture of their root systems and thereby root characteristics such as branches and branch endings are analysed. The root of the small plant *Arabidopsis thaliana* is excellently suited for large-scale non-invasive phenotyping because it can be grown on transparent media in large numbers and projections of the young root essentially capture all the important biological features at the organ level [25]. For analysis of the plant development, the plants are imaged on several successive days of their growth cycle.
The root characteristics used in plant phenotyping, for example branching points or length of branches, can be captured and described by a graph based shape representation.
Branches in the root structure depict a change in the topology of the structure. Reeb graphs describe these changes in topology in the represented structure. As the 3D structure of the root is projected to the 2D image space, overlaps of branches may occur in the root images. These overlaps need to be distinguished from branching points. One of the advantages in using Reeb graphs is that such overlaps can immediately be detected, as they introduce a particular type of node in the graph. Reeb graphs are based on Morse theory and analyse the image content according to a function (Morse function). An important question in this context is to determine how much and where two or more functions defined on the same model differ [6]. A combined graph representation overcomes these differences of various representations and allows to combine the advantages of multiple representations.
In the context of plant phenotyping, a combined representation is needed to compare plants or their development based on various characteristics and thereby various representations.

## 1.2 Aim of the Work

In order to simplify the examination of root characteristics and enable an efficient comparison of roots, a representation of imaged root data by Reeb graphs is introduced in this thesis.
Reeb graphs capture the topology of the represented structure - in this case the locations of branches and branch endings of the roots - and form a skeletal graph representation of the underlying image data in this way. The computation of the Reeb graphs on the root images is based on a Morse function. For the root images a Reeb graph representation based on the height function and the geodesic distance is presented. The medial axis transformation was applied for a third representation. Based on these three representations the following characteristics of the roots can be acquired:

- number of branches,
- number of branch endings,
- length of branches,

- width of branches,
- branching pattern (according to a main stem).

To allow for an efficient comparison of different plants or of one plant in different stages of its development, a normalised representation is defined.
Graphs provide a representation that allows to compare and match shapes. However, for such comparisons it is usually assumed that all graphs were computed based on the same method. In order to combine characteristics based on all three mentioned graph representations, to obtain the final combined representation, the similarity of graphs based on different functions is measured.

## 1.3 Contribution

Based on three graph based shape descriptors an approach to represent root structures is described in this thesis. The medial axis is a popular shape descriptor for branched structures such as roots. In this thesis a representation of branched structures based on Reeb graphs is introduced. In order to combine characteristics of different representations, the equality of various types of representations is evaluated. A definition for the comparison and matching of two different graph representations forms a central part of this work. For the efficient comparison of root images of different plants or of one plant of different days of growth, a normalised graph representation is developed. The derived root representations are compared to the representations computed with the 2014 published tool "Root System Analyser" [31].

## 1.4 Structure of the Thesis

An overview on graph based shape representations in general and the theoretical background particularly needed for the introduction of Reeb graphs are given in Chapter 2. Based on this knowledge the state of the art regarding (Reeb) graph based representations as well as the state of the art in image based plant phenotyping are discussed in Chapter 3. Chapter 4 presents the dataset used within the scope of this thesis. Details on the presented approach for Reeb graph based root representations are given in Chapter 5, while results are discussed in Chapter 6. A comparison with the recently published tool "Root System Analyser" [31] is given in Section 6.3. The possibility of a combination of the different representations is discussed in the Sections 6.4 - 6.6. Chapter 7 concludes this thesis and gives an outlook to future work.

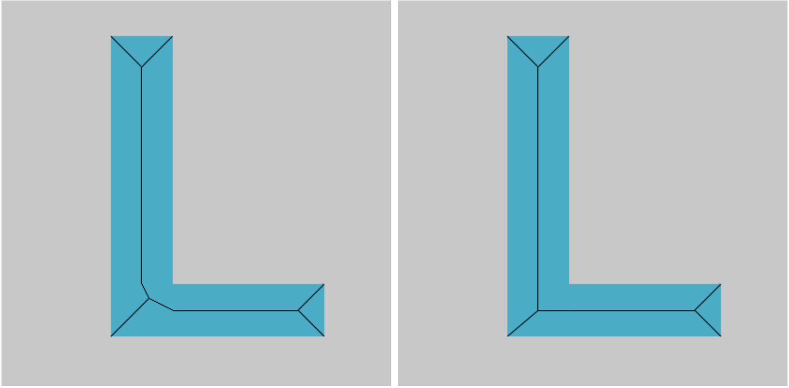
CHAPTER 2

# Graph Based Shape Representation

For the purpose of graph based shape representation, fundamentals of skeletons and graphs are needed. The following sections provide the theoretical basis for the concepts presented in this thesis.

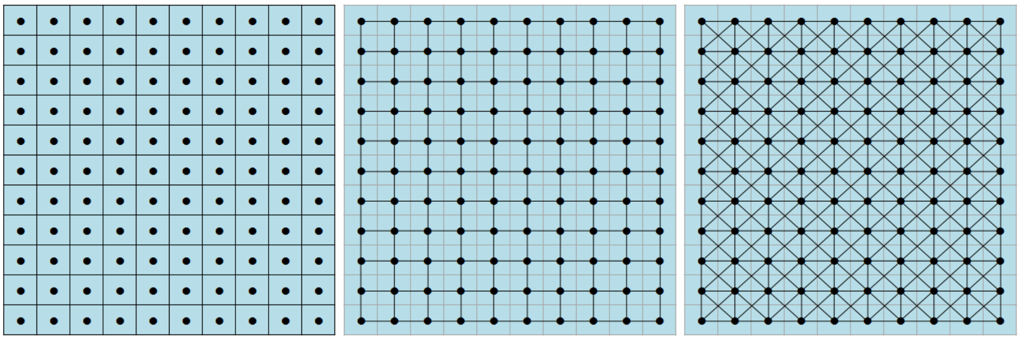
## 2.1 Graphs and Skeletons

In shape representation a skeleton is a thinned version of a shape with equal distance to the boundaries of the shape. Morphological skeletons, known as medial axis, are the most common skeletons. Sometimes the terms skeleton and medial axis are used interchangeably as synonyms. However, Figure 2.1 presents a comparison of two different types of skeletons. Figure 2.1a shows an example for a medial axis skeleton, while Figure 2.1b shows another type of skeleton: a straight line skeleton.



(a) medial axis skeleton (b) straight line skeleton

Figure 2.1: Shape representation: skeleton.

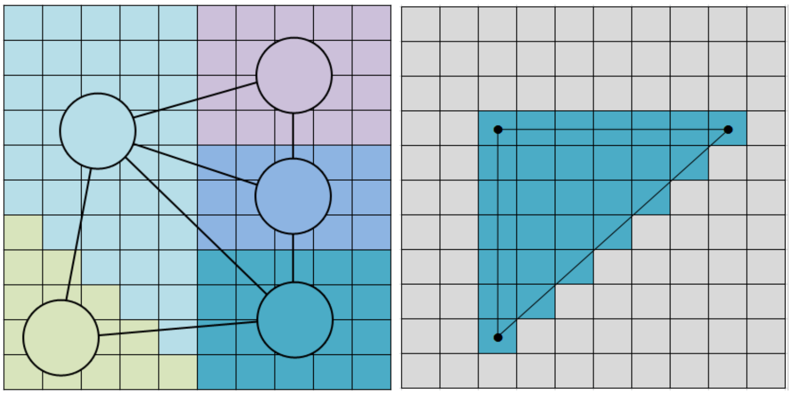


(a) 10x10 pixel image - one node per pixel (b) graph based on 4-connected grid (c) graph based on 8-connected grid

Figure 2.2: Graphs on image pixels.

**Definition 1** *A graph $G$ is defined as a tupel $(V, E)$ where $V$ is the set of nodes (vertices) of the graph and $E$ the set of edges connecting the nodes in the graph ($E \in V \times V$). Edges in a graph may be directed or undirected [18].*

A digital image is formed by a set of pixels. A graph on such an image can be defined by mapping a node to each pixel and connecting nodes of neighbouring pixels with edges. Figure 2.2 shows an example of such a graph assuming a 4-connected pixel grid and an 8-connected one. For graph-based image representations nodes may not be assigned to every pixel in the graph. Pixels are rather grouped and each group is represented by a single node in the graph. The arrangement of pixels into groups can be based on similarity of an assigned label or value as for example their colour-value. An example for such a graph is shown in Figure 2.3a. This graph is a region-adjacency-graph (RAG): each region is represented by one node, the edges represent adjacency between regions. In such a way graphs can for example be used in segmentation ap-



(a) RAG - nodes represent regions, edges their adjacency (b) representation of shape - nodes describe characteristic locations

Figure 2.3: Two examples for graph based image representation.

proaches.
For the description of shapes, nodes are placed at characteristic positions of a shape and connected accordingly to it. The triangle image in Figure 2.3b is represented by 3 nodes at the vertices of the triangle and edges according to the triangle edges.

**Definition 2** *A graph contains a cycle if there exists a subset of edges in the graph, that forms a path that starts and ends at the same node [18].*

The graph for the triangle in Figure 2.3b for example forms a cycle. The number of nodes, edges and cycles in a graph are linked to each other through the Euler characteristic:

**Definition 3** *The number of edges in a graph equals the number of nodes in the graph minus one, plus the number of cycles in the graph. For a graph $G = (V, E)$ with cycles $C$ the number of edges $|E|$ is given as: $|E| = |V| - 1 + |C|$ [18].*

### Operations on graphs

To remove or insert nodes or edges in a graph, edit operations are defined for graphs. These modifications on a graph may change to connectedness of this graph.

**Definition 4** *A graph $G = (V, E)$ is called connected if, for any pair of nodes $p \in V$ and $q \in V$, there exists a path from $p$ to $q$ . Otherwise this graph is said to be disconnected [18].*

**Definition 5** *Edit operations for graphs based on [18]:*

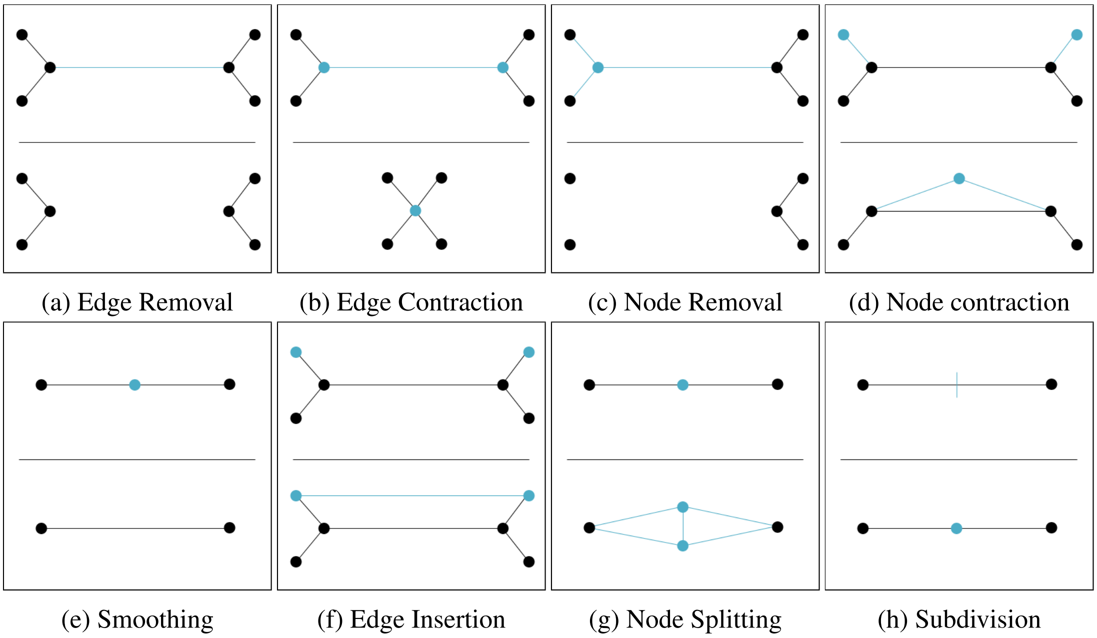
1. *Edge Removal:*
   *deletes an edge from the graph, while keeping the endpoints of the graph unchanged in the set of nodes (see Figure 2.4a).*

2. *Edge Contraction:*
   *removes an edge from the graph by merging the two nodes that were connected by the edge (see Figure 2.4b).*

3. *Node Removal:*
   *deletes a node from the graph, all edges containing this node are removed from the graph as well (see Figure 2.4c).*

4. *Node Contraction:*
   *merges two nodes of a graph to one node. The adjacency of this new node is given as the union of the sets of adjacent nodes of the two merged nodes. A restriction, that the two nodes need to be connected by an edge, does not necessarily apply. Edge contraction therefore is a special case of node contraction (see Figure 2.4d, respectively see Figure 2.4b).*

5. *Smoothing:*
   *removes a node from the graph by removing the edges connecting it to other nodes and establishing edges between these nodes (see Figure 2.4e). This operation is only possible for*



(a) Edge Removal (b) Edge Contraction (c) Node Removal (d) Node contraction

(e) Smoothing (f) Edge Insertion (g) Node Splitting (h) Subdivision

Figure 2.4: Operations on graphs. The top half shows the original graphs with changing parts labelled blue, the bottom half of the images shows the altered graphs.

*nodes of degree two (nodes adjacent to two other nodes). The difference to edge contraction lies in the node that is kept when applying this operation. Edge contraction applied to the graph in Figure 2.4e would remove one of the degree-1 nodes, while smoothing removes the degree-2 node. This is particularly important, when the position of nodes is needed to further represent the geometry of the structure, the graph is based on.*

6. *Node insertion:*
   *adds a node to the set of nodes of the graph. This node is not connected to any other nodes of the graph, edges need to be inserted (see edge insertion) in order to connect this node to other nodes.*

7. *Edge insertion:*
   *establishes an edge between two nodes in the graph (see Figure 2.4f).*

8. *Node Splitting:*
   *splits one node into two nodes, which are connected by an edge. Both nodes are adjacent to all nodes the original node was adjacent to (see Figure 2.4g).*

9. *Subdivision:*
   *divides an edge into two edges by introducing a new node that is adjacent to the two nodes of the original edge (see Figure 2.4h). Subdivision is the reverse operation to smoothing.*

### Subgraphs and Graph Isomorphism

An equivalence relation on graphs is given by graph isomorphism. If only parts of two graphs are considered equivalent, the equivalence relation is defined for this parts of the graphs as subgraph isomorphism.

- Graph isomorphism:

  **Definition 6** *Graph isomorphism is a bijection between the sets of nodes of two graphs $G_1$ and $G_2$, such that for each edge in $G_1$ there exists an edge in $G_2$.*

  The bijective function $f : G_1 \mapsto G_2$ is called isomorphism [8]. The two graphs in Figure 2.5a are isomorphic graphs, corresponding nodes are coloured the same way.

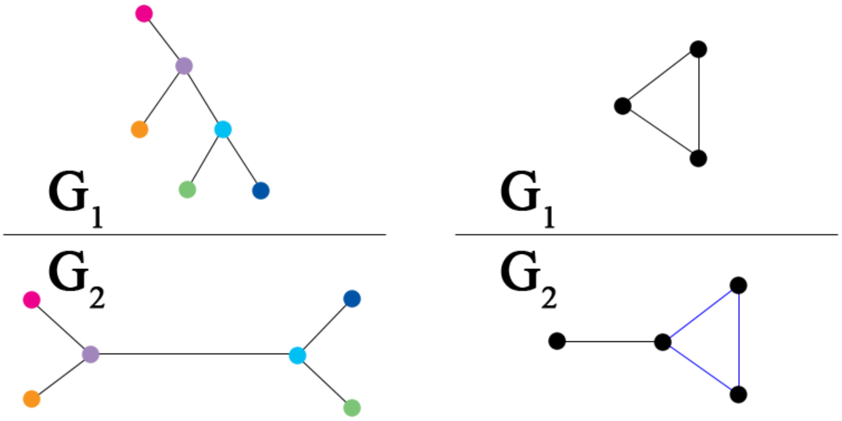
- Subgraphs:

  **Definition 7** *The node set of a subgraph $G' = (V', E')$ of a graph $G = (V, E)$ is a subset of the node set of $G$. The edge set of the subgraph $G'$ is the subset of the edge set of $G$ restricted to the nodes in $V'$: $V' \subseteq V$ and $E' = E \cap (V' \times V')$ [8].*

  Figure 2.5b shows graph $G_1$ as a subgraph of $G_2$, the corresponding edges of the subgraph are coloured in blue.

- Subgraph isomorphism:

  **Definition 8** *A subgraph isomorphism from $G$ to $G'$ is the injective function on the two graphs $G = (V, E)$ and $G' = (V', E')$: $f : G \mapsto G'$ if there exists a subgraph $S \subseteq G'$ so that $f$ is a graph isomorphism from $G$ to $S$ [8].*



(a) $G_1$ and $G_2$ are isomorphic graphs

(b) $G_1$ is a subgraph of $G_2$ (edges of the subgraph are marked in blue)

Figure 2.5: Definition of subgraphs and graph isomorphism.

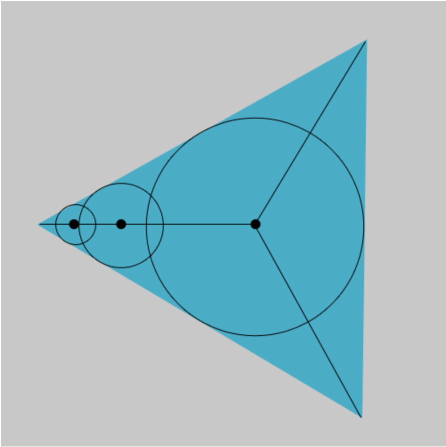


Figure 2.6: Computation of a medial axis by inscription of maximal circles.

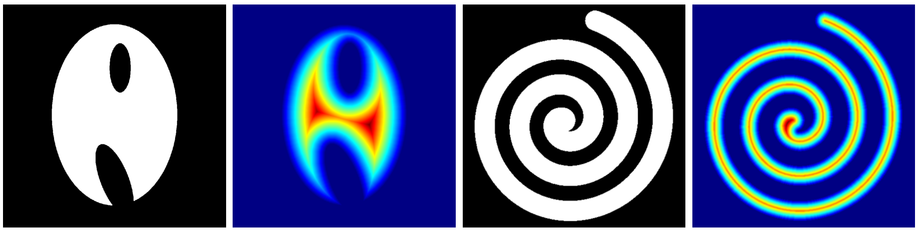
In Figure 2.5b there is a subgraph isomorphism from $G_1$ to $G_2$.

## 2.2 Medial Axis Transformation

The medial axis transformation (MAT) was introduced by Blum in 1967 [19] and has since been widely used in shape analysis and representation. A skeleton (the medial axis) is computed on a shape.

**Definition 9** *For a shape $S$ (for example a polygon) the medial axis $M(S)$ is the set of points $\{p\}$ inside $S$, for which at least two points in the boundary of $S$ are closest and equidistant to $\{p\}$ [30].*

The skeleton, derived based on the MAT, is a symmetric axis of the shape.
Using a geometric description: the medial axis is formed by the centres of maximal circles that are inscribed into the shape at any position. Figure 2.6 shows an example. Therefore the MAT further provides a measurement of width, as for each point on the medial axis, a radius



(a) shape 1 (b) distance transformation (c) shape 2 (d) distance transformation

Figure 2.7: Distance transformation computed for two shapes. Red indicates high distance values, blue indicates low distance values.

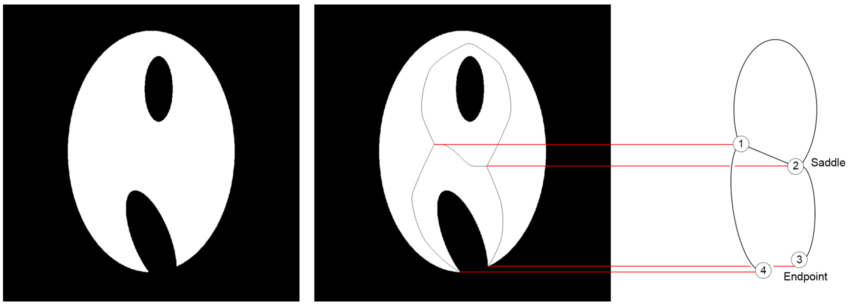


Figure 2.8: Graph representation of a shape derived based on the medial axis skeleton.

(the distance to the boundary of the shape) is associated with this point [30]. The shape can be reconstructed based on the medial axis and the width information stored with each skeleton point.

A distance transformation on a shape labels each foreground pixel with the distance to the next background pixel. The medial axis corresponds with the ridges formed by the local maximum values in the distance transformation. Figure 2.7 shows the values of a distance transformation computed for two shapes.

The medial axis skeleton can be used as an intermediate step to build a graph representation of a shape. In order to compute a graph based on the skeleton, branching points and endpoints in the skeleton need to be detected. These are used as the nodes in the graph. For the medial axis graph the nodes are connected by edges according to the skeleton. The medial axis skeleton computed on a shape, along with the corresponding graph, is shown in Figure 2.8.

## 2.3 Morse Theory and Reeb Graphs

Reeb graphs are compact shape descriptors that preserve the topological characteristics of the described shape [6]. Reeb graphs are named after the French mathematician Georg Reeb and are based in Morse theory [7]. Morse theory provides an analsis of the topology of a manifold. The topological characteristics of interest are the number of connected components and the number of holes in the 2D structure. These two characteristics are the first two Betti numbers $b_0$ and $b_1$. $b_0$ is defined as the number of connected components of a topological space and $b_1$ as the number of its 2-dimensional holes [22].

Based on critical points according to a scalar function a Reeb graph describes the topological structure that is the connectivity of level sets of e.g. 2D or 3D content [12]. In order to build a Reeb graph, critical points, of the structure to be represented, need to be computed.

**Definition 10** *A point $(a, b)$ of a function $f(x, y)$ is called a critical point if both derivatives $f_x(a, b)$ and $f_y(a, b)$ are equal to $0$ or if one of these partial derivatives does not exist [41].*

Such a critical point can either be degenerate or non-degenerate, an example in 1D for both types of critical points is given in Figure 2.9. These two cases can be distinguished via the Hessian matrix for a twice differentiable function. The determinant of the Hessian matrix at a critical

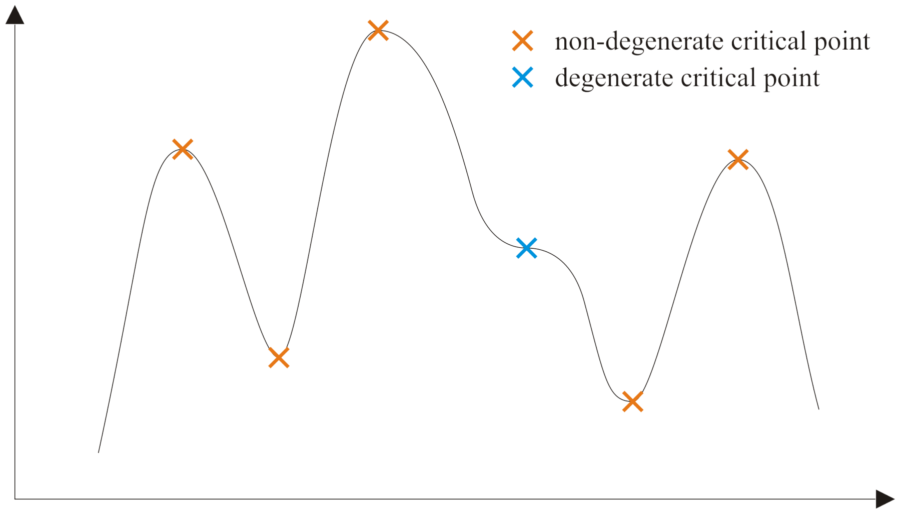


Figure 2.9: Degenerate and non-degenerate critical points of a function.

point $x$ is then called the discriminant. If this determinant is zero then $x$ is called a degenerate critical point of $f$ (or non-Morse critical point of $f$). Otherwise, it is non-degenerate (or Morse critical point of $f$).

**Definition 11** *A smooth, real-valued function $f : M_d \rightarrow \mathbb{R}$ is called a Morse function if it satisfies the following conditions ($M_1$ - $M_3$) for a $d$ manifold $M_d$ with or without boundary:*

- *$M_1$: all critical points of $f$ are non-degenerate and lie inside $M_d$,*
- *$M_2$: all critical points of $f$ restricted to the boundary of $M_d$ are non-degenerate,*
- *$M_3$: for all pairs of distinct critical points $p$ and $q$, $f(p) \neq f(q)$ must hold [11].*

Critical points of such a real-valued function are those points where the gradient becomes zero. The topological information of a shape described by a Reeb graph based on a function is related to the level sets of this function on the shape [6]. A change in topology appears with a change in the number of connected components in a level set. At regular points no topology changes occur. Topological changes occur at critical points only.
Nodes of the Reeb graph correspond to critical points of the function (points where the topology changes), edges describe topological persistence [6]. In other words: All nodes having the same function value are represented by one node in the graph, connections between nodes describe connections between segments of the underlying structure. A first example for such a Reeb graph is shown in Figure 2.10. The white foreground region is here analysed top down. With a change in the number of connected components at a certain height a node is introduced in the graph.
Reeb graphs are originally defined for the continuous space, but have been extended to the discrete domain: Here the Reeb graph is defined on a piecewise linear Morse function [12]. As the approach presented in this thesis provides an analysis of 2D image content, it is based in the

discrete domain (image pixels). The Reeb graphs that are built on the root images are therefore discrete Reeb graphs and are based on the definitions below.

In order to define a discrete Reeb graph, connective point sets and level-set curves are defined first:

**Definition 12** *Two point sets are connected if there exists a pair of points (one point of each point set) with a distance between these two points below a fixed threshold [44].*

**Definition 13** *If all non-empty subsets of a point set, as well as its complements, are connected, such a point set is called connective [44].*

**Definition 14** *A group of points that have the same Morse function value and that form a connective point set, is called a level-set curve [44].*

Based on the Definitions 12 - 14 a discrete Reeb graph is defined as follows:

**Definition 15** *The nodes in a discrete Reeb graph represent level-set curves, the edges connect two adjacent level-set curves, therefore the underlying point sets are connected [44].*

The critical points are computed on the shape according to a Morse function. The most popular function in this scope is the height function. But distance functions as for example the geodesic distance are suitable as well.

**Definition 16** *The height function in 2D is defined as the function $f$ that associates for each point $p = (a, b)$ of a function $f(x, y)$ the value $b$ as the height of this point $p$: $f(x, y) \mapsto y$.*

**Definition 17** *The geodesic distance is defined as the shortest distance measured between two points.*

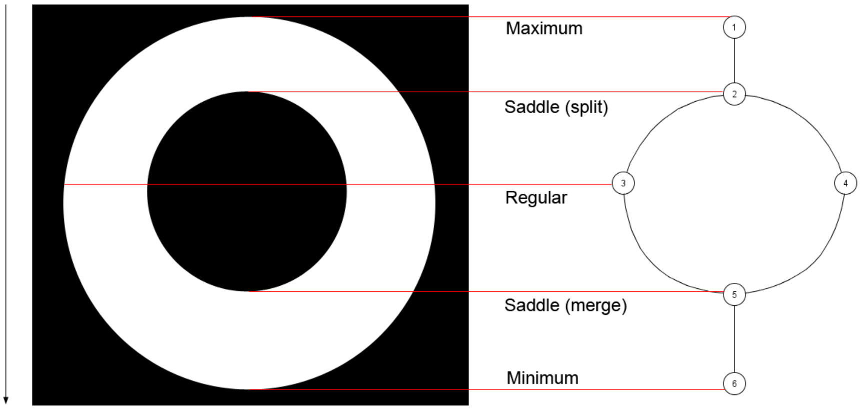
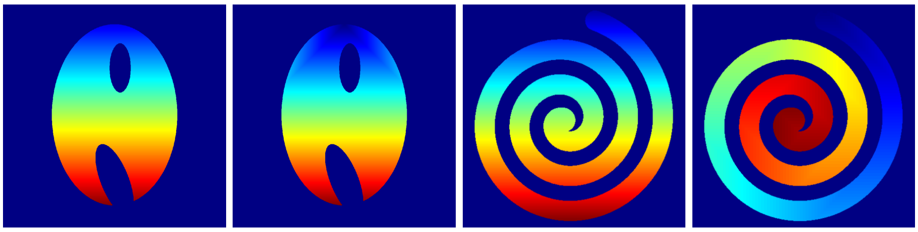


Figure 2.10: Critical points computed based on the height function (downwards) and corresponding Reeb graph. The white image region shows the foreground region described by the Reeb graph, black parts are background.



(a) height function (b) geodesic distance (c) height function (d) geodesic distance

Figure 2.11: Example images for the two Morse functions: the height function is computed top-down, the source point for the geodesic distance is in the centre of the topmost image row of the foreground, the chessboard distance is used as metric here. The input images are shown in Figure 2.7a and 2.7c. Red indicates high function values, blue low function values.

Within the scope of this thesis the geodesic distance is measured as the chessboard distance between any point of the root and the starting point of the root which is located at the centre pixel of the topmost foreground image row.

A comparison of the function values generated by these two Morse functions is shown in Figure 2.11. For Figure 2.11a and 2.11b the function values are very similar, as are their Reeb graphs (shown in Figure 2.12). The Figures 2.11c and 2.11d show an example where the function values for height function and geodesic distance as well as the according Reeb graphs vary strongly.

In 2D critical points and corresponding nodes in the Reeb graph are minima, maxima or saddles [11]. The saddle nodes can be further distinguished: a saddle node that appears with a reduction in the number of connected components is further called merge (saddle) node, a split (saddle) node describes an increase in the number of connected components. When considering these two different types of saddle nodes that might appear in a Reeb graph, four different types of critical points and according nodes in the graph can be distinguished.

**Definition 18** *The different types of nodes for a Reeb graph in 2D are: maximum node, minimum node, split (saddle) node and merge (saddle) node.*

Within the scope if this thesis, a maximum node is defined as the birth of a new connected component, while a minimum node describes the death of a connected component.

Besides these nodes corresponding to critical points, regular nodes can be added at any position and along any edge in the Reeb graph, as they do not describe a change in topology. Nevertheless, regular nodes can, for example, be used to describe changes in the colour of the foreground region (see [5]).

**Definition 19** *A node of type minimum or maximum has degree one (it is adjacent to one node in the graph), saddle nodes have degree three (they are adjacent to three nodes) while regular nodes have degree two (adjacent to two other nodes).*

Figure 2.10 shows an example for a Reeb graph based on a height function, containing all five types of nodes and the actual image, the graph was computed on. Each edge in the Reeb graph

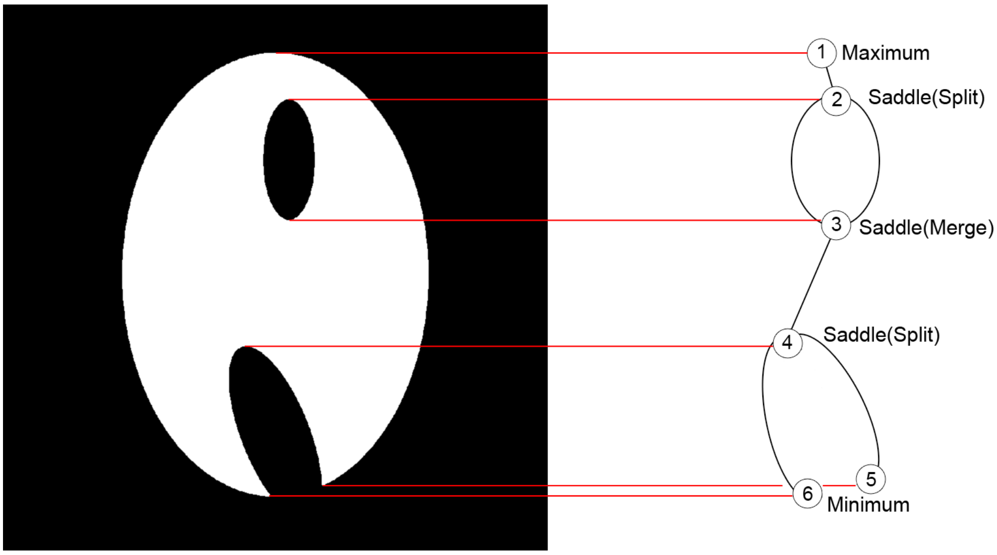


Figure 2.12: Reeb graph, computed for the white foreground region, according to the height function (going downwards) and the geodesic distance (the source pixel is located in the centre of the top foreground image row). Here both Morse functions generate identical critical points.

describes a connected component. Therefore, the edges of a Reeb graph are formed by connecting the node representing the birth of a connected component to the corresponding node representing the death of this component.
Another example Reeb graph containing all possible types of nodes is shown in Figure 2.12. The nodes in this graph correspond to the critical points of two Morse functions: the height function (going downwards) as well as the geodesic distance (the source pixel is located in the centre of the top foreground image row) both result in this set of nodes.

The Reeb graph not only depends on the shape it represents but as well on the Morse function, used to compute the critical points. While the height function is easy to compute, it is not rotational invariant [4]. A geodesic distance from a source point provides a Morse function invariant to rotations, as long as the source point is set to the same position inside the shape. Height function and geodesic distance as Morse functions produce equivalent Reeb graphs, in case the maximum point (source point) of the geodesic distance is at the same position as maximum node according to the height function and changes in the topology of the represented shape occur only in a vertical direction (see Figure 2.12).

CHAPTER 3

# State of the Art

This thesis addresses two main areas of research: image based plant phenotyping and graph based shape representations.

## 3.1 Image Based Plant Phenotyping

In the research area of plant phenotyping the (semi-)automatic extraction of traits from image data allows for an analysis of large numbers of individuals. Table 3.1 shows an overview of available applications [1] that allow for a description of root systems or measurements on the roots. A very recent approach to describe root structures based on a skeleton representation of 2D image data and a 3D root reconstruction that allows for comparisons of root systems is discussed in the following sections.

### Root System Analyser

The 2014 published tool “Root System Analyser” [31] recovers root architectural parameters based on 2D images of the roots. The root system is represented as a graph in this approach. The approach proceeds based on the following operations:

1. Skeleton representation:
   The root image is transformed into a medial axis skeleton in a first step.

2. Graph representation:
   The branching points and end points are detected in order to form the nodes of the graph. As neighbouring nodes are connected in the graph, an adjacency matrix is created, that stores this neighbourhood information. The coordinates of edges are stored in an edge-list. Figure 3.1 shows four different root input images, the colour indicates the age of the

[1] http://www.plant-image-analysis.org/

Table 3.1: Available root system analysis and measurement applications.

| Root system analysis tools | | |
|---|---|---|
| **Name** | **Analysis / measurements** | **Published in** |
| DART | length, insertion, topology | [29] |
| EZ-Rhizo | length, insertion, topology, insert.-angle, #branches | [1] |
| GiA Roots | convex hull, length, perimeter, surface, volume, #branches, depth | [14] |
| GrowScreen-Root | length, insert.-angle, #branches | [35] |
| IJ-Rihzo | length, diameter | [37] |
| RootFly | length, diameter, colour | [45] |
| RootNav | length, count, convex-hull, insertion, insert.-angle | [38] |
| RooTrak | 3D reconstruction | [33] |
| RootReader 2D | #branches, length, topology, depth | [9] |
| RootReader 3D | length, width, depth, volume, surface, convex-hull, #branches, orientation, insert.-angle | [10] |
| RootScape | shape | [39] |
| Root System Analyser | length, insertion, insert.-angle, diameter, count | [31] |
| RootTrace | length, curvature, #branches | [13] |
| SmartRoot | diameter, insertion, insert.-angle, length, orientation, #branches, topology | [32] |
| WinRhizo / WinRhizo TRON | colour, diameter, length, topology, volume, surface | [2] |

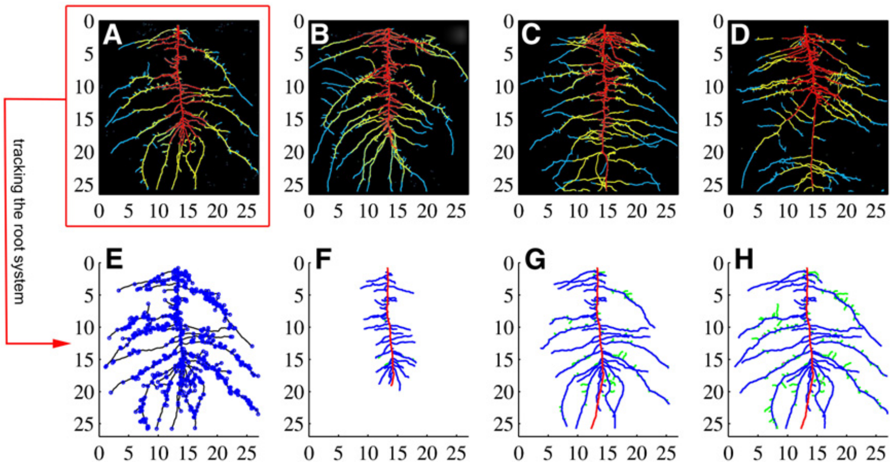
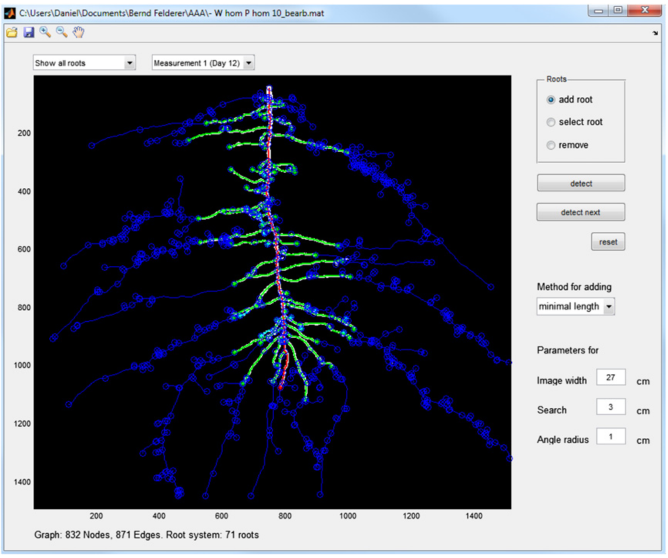


Figure 3.1: Images A-D show root input images, colours indicate the age of the root, with red labelling the oldest parts of the root. The corresponding graph representation of image A is shown in image E, individual branches of the root system are marked in the images F-H. Image taken from [31].



C:\Users\Daniel\Documents\Bernd Felderer\AAA\- W hom P hom 10_bearb.mat

Show all roots

Measurement 1 (Day 12)

Roots
● add root
○ select root
○ remove

Method for adding: minimal length

Parameters for

Image width 27 cm

Search 3 cm

Angle radius 1 cm

Graph: 832 Nodes, 871 Edges. Root system: 71 roots

Figure 3.2: User interface of the “Root System Analyser”. Image taken from [31].

root, with red parts being the oldest parts of the root. For image A the nodes forming the graph are shown in image E.

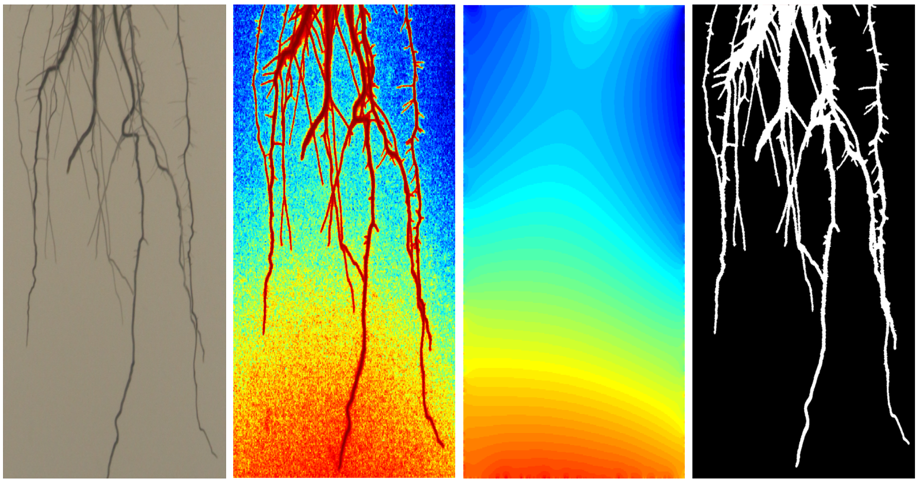
3. Root tracking:
   To determine individual branches, an underlying dynamic root architecture model is applied. The algorithm starts at predefined tips of roots and estimates the growth of the root for a small time step. All possible paths for this time step are found in the graph. The optimal growth path is determined dependent on the following characteristics: straightness and average diameter and it is penalised if an edge is already assigned to a root. Based on this method, overlaps and branching points can be distinguished. Figure 3.1 shows the marked individual branches of input images B-D in the images F-H.

A graphical user interface allows the user to select individual branches, characteristics of these branches as for example length or width of the branch are then displayed. Moreover the user is able to correct the root system to a certain degree, by deleting roots or adding roots. Figure 3.2 shows an example for this graphical user interface.

### Reconstruction of 3D Plant Root Shape

A related work on the analysis of branching patterns of roots based on a 3D reconstruction of the root architecture of rice plants is provided in [46].The aim of this work is to build a 3D model of the branching pattern of roots based on 2D images. This 3D model allows for a comparison of the root systems of plants with different genotypes. The approach progresses in three steps:

1. Harmonic Background Substraction:
   For the separation of the foreground (the root) from the background (the rest of the im-



(a) grayscale input root image (b) normalised intensities (c) harmonic background model (d) segmented root image

Figure 3.3: Segmentation process to separate foreground (root) from background. Image taken from [46].

age) the background is modeled according to a harmonic function. The foreground is constructed using the difference between an image and the background model. Figure 3.3 illustrates this segmentation process. The authors showed that this approach more reliably preserves fine root structures than a single threshold or hysteresis thresholding.

2. Regularised Visual Hull:
3D shapes are often reconstructed from 2D images by the visual hull method. In this approach the visual hull is extended by a regularisation term to a regularised visual hull. The visual hull can be defined as the set of voxels that maximises the total consistency. When $v \in \pi_k^{-1}(F_k)$ is the maximal set of voxels with projection $F_k$ ($F_k$ is the foreground in the $k$-th image):

> "Define the consistency of a voxel $v$ with the $k$-th image as:

$$cons_k(v) = \begin{cases} 1 & \text{if } v \in \pi_k^{-1}(F_k), \\ \text{-N} & \text{otherwise.} \end{cases} \tag{3.1}$$

and its total consistency as $cons(v) = \sum_{k=1}^{N} cons_k(v)$. Then the visual hull is the set of voxels that maximises the total consistency:

$$V = arg\,\max_S \sum_{v \in S} cons(v)\text{." [46]} \tag{3.2}$$

Additionally, one of the root images is used to improve the 3D reconstruction based on a regularisation parameter $\lambda \geq 0$. This one image guides the 3D reconstruction dependent on $\lambda$. If $\lambda$ is small, the influence of this image is low. The standard visual hull $V$ is computed and taken as a first regularised visual hull $V_\lambda$. Each pixel in the foreground of the "regularisation image" $F_i$ is visited next. If pixel $u$ is not covered, a voxel with maximal consistency measure $cons(v)$ in the set of all voxels with projection $F_i$ is found. $cons(v)$ is negative here as $u$ is not covered yet. If $cons(v) + \lambda$ is positive, $v$ is added to $V_\lambda$, otherwise $v$ is discarded.

3. Repairing Connectivity:
   Two voxels are connected if they share a 2-dimensional face. As the regularised visual hull can consist of more than one connected component, the connectivity is repaired in a final step of the reconstruction. This is done by solving an optimisation problem: as a solution a connected set of voxels $U$ is required that minimises the maximum distance to $V_\lambda$ and the minimum inconsistency with the 2D images. This is done by building a graph that describes the connectivity of the voxels in the 3D reconstruction. The weights of the edges are set as the larger inconsistency value of the two nodes of the edge. Based on this, a minimum spanning tree is computed. The minimum cost path describes the path between two voxels that minimises the weight of the edges in between. A voxel that lies on such a path is called a separating voxel. The solution to the optimisation problem consists of all voxels in $V_\lambda$ and all separating voxels of the minimum spanning tree.

## 3.2 Graph Based Shape Description

Skeletons and graph structures are used for shape representation and comparison.
Since its introduction in 1967 the medial axis has been used in a multitude of applications and developed into a standard approach for a simple shape representation. While the medial axis provides a skeleton representation that can serve as a basis for a graph representation, Reeb graphs derive a graph directly on the image data (as shown in [15]). For the representation of 3D data Reeb graphs can be efficiently derived from meshes (as shown in [44]) or from point clouds (presented in [36]). The following sections discuss a Reeb graph based approach to derive a skeleton representation based on a point set as input data as well as an approach for skeletonisation of 3D scan data. Both approaches use the geodesic distance as a basic function to compute the Reeb graph. Besides the height function, the geodesic distance is as well used for the Reeb graph computation in the approach presented in this thesis.

### Data skeletonisation via Reeb graphs

The basic idea of the approach presented in [15] is to find a hidden space in discrete samples that has a graph-like geometric structure. A skeleton graph is computed based on a point set as input data. An example of a skeletonisation of a 2D image of a Chinese character obtained through this approach is shown in Figure 3.4. The proposed method proceeds as follows:

1. Simplicial complex K:
   The hidden domain, the input points are sampled from, needs to be approximated. This is done using a simplicial complex K. First a proximity graph is constructed. All input points

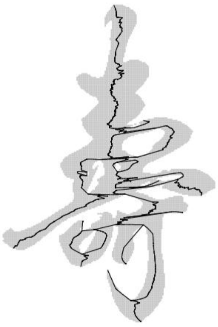


Figure 3.4: Reeb graph based skeletonisation result of the input data in gray. Image taken from [15].

are therefore connected to their neighbours based on either the $k$-nearest neighbourhood (the $k$ neighbours that are closest to a point) or the $r$-neighbourhood (the neighbours within distance $r$ from a point) definition. All points and edges from this proximity graph are added to the simplicial complex. For any three points that are pairwise connected in the proximity graph, a triangle is added to the simplicial complex.

2. Reeb graph computation:
   The Reeb graph is computed on the simplicial complex K using the geodesic distance according to a base point $b \in$ K. $b$ is obtained by taking an arbitrary point $v \in$ K and defining $b$ as the point furthest away from $v$. The geodesic distance function is here approximated by the shortest distance in the proximity graph. The (augmented) Reeb graph is computed according to the approach described in [20]: for a point $q$ and the connected component $C_q$ that holds $q$, all triangles of the simplicial complex that intersect $C_q$ are collapsed onto $q$ and its adjacent edges. This procedure is repeated for all vertices in K.

3. Post-processing:
   In case the represented data was not already embedded in 2D or 3D, the input points are projected to $\mathbb{R}^3$. The projected points are connected according to the Reeb graph connectivity and are iteratively smoothed by substituting the position of a point by the

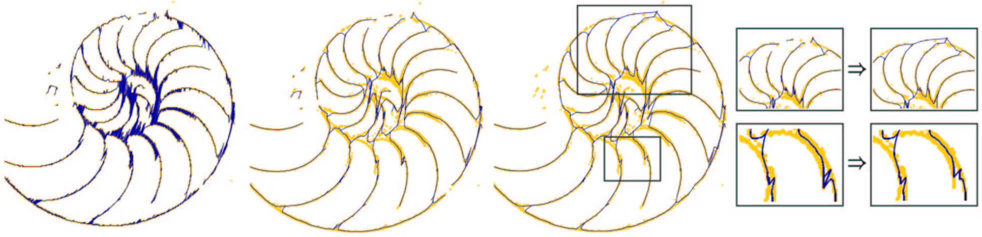


Figure 3.5: Reeb graph based skeletonisation (left to right): input points in yellow - augmented Reeb graph - iterative smoothing - post-processing: adding missing links (top) and removing spurious branches and loops (bottom). Image taken from [15].

average of its neighbours. Missing links are added by connecting pairs of degree-1 nodes with a distance smaller than a certain threshold. To remove spurious branches or loops in the Reeb graph features and their persistence are computed on the graph. The graph is simplified by merging features with a persistence smaller than a predefined threshold.

Figure 3.5 shows example images for the methodological sequence described above.

### Human body scan segmentation based on Reeb graphs

The overall goal of the approach presented in [44] is to segment a 3D human body scan into subsets which correspond to functional body parts. As this segmentation is done pose-independent, a graph representation is computed on the 3D data first. The approach is divided into the following operations:

1. Computation of Morse functions:
   For the computation of the Reeb graph two Morse functions are used: the geodesic distance between a point and a source point and the sum of geodesic distances. The sum of geodesic distances for one point is computed as the sum of the geodesic distances to all other points of the surface. The geodesic distance to a source point is computed on the voxels using a wavefront approach: starting from a source point in each iteration all 26 neighbouring voxels are taken as a level set with the same distance to the source. In this way the level-sets are already extracted while computing the Morse function. The sum of geodesic distances requires the computation and summation of the geodesic distance for all voxels with the current voxel as source point.

2. Decomposition into level-set curves:
   Based on the extracted level-sets the Reeb graph is constructed by representing each level-set curve, a iso-valued connective point set, by a node in the graph. The edges in the Reeb graph are formed according to the connection of level-set curves with each other and connect nodes of two adjacent level-set curves.

3. Extraction of branches:
   In order to extract branches corresponding to body parts, critical nodes in the Reeb graph need to be detected. The main problem is that noise in the data leads to false positives, critical nodes that only describe noise. Critical nodes are divided into three types of nodes: O-type (loop), $\lambda$-type (split) and Y-type (merge). While the O-type consist of two saddle nodes connected by two edges, the types $\lambda$ and Y are both consistent of one saddle node connected to two leaf nodes. While these two types are topologically equivalent, they can be distinguished when taking the direction of the Morse function into account. The human body topology cannot produce O-type saddle nodes. Based on the direction of the Morse function one of the two saddle nodes $\lambda$-type and Y-type cannot occur for a human body as well: According to the geodesic distance, there is only one maximum node (the source voxel), all other nodes are saddle nodes of type split or minimum nodes. These nodes are therefore caused by data corruption and can be discarded. Furthermore, it is assumed that a branch associated to a true either $\lambda$-type or Y-type saddle node may

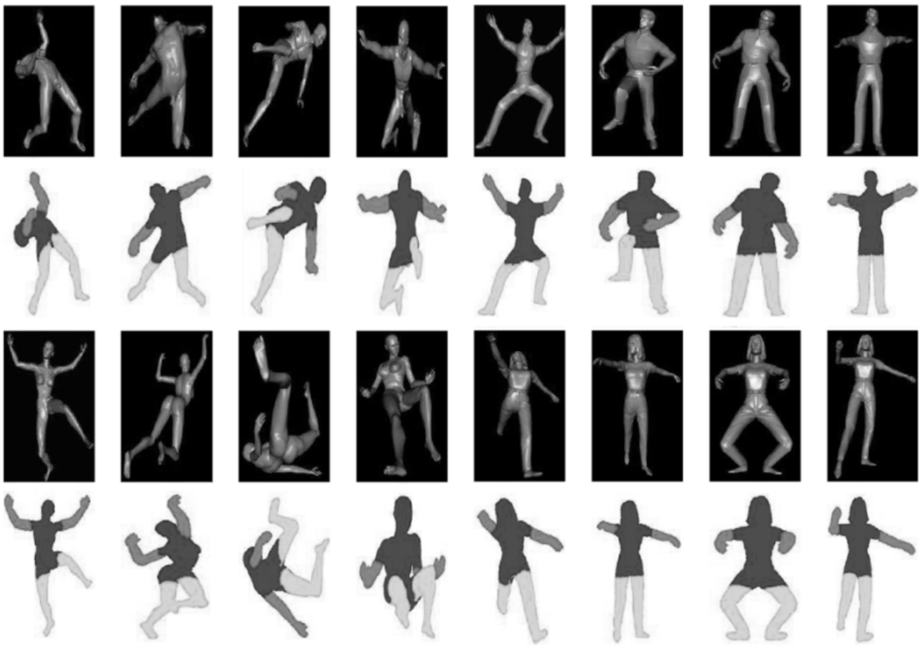


Figure 3.6: Segmentation results of simulated scans. Image taken from [44].

not be shorter than the length of the smallest body part. In this way false critical nodes are discarded. The experiments in [44] show the robustness of the approach considering artefacts as O-type saddle nodes, or added Gaussian noise of different amplitudes. True critical nodes are detected: these nodes represent the bottom of the feet, the groin, hand tips, armpits and top of the head. Body parts are identified based on the branches between the nodes and the 3D data is segmented into legs (branches between feet and groin) and arms (branches between hands and armpits). The rest of the data corresponds to the torso and head segment.

Figure 3.6 shows the final segmentation for synthetic 3D data.

CHAPTER 4

# Dataset and Pre-Processing

The images of the root dataset were acquired by the Gregor Mendel Institute of Molecular Plant Biology (GMI). For this dataset images of the plant *Arabidopsis thaliana* were taken. This plant is a model organism, which is widely used in plant sciences, due to the small size of its genome, the small size of the plant itself and its rapid life-cycle [21].

The whole set of plant images used here consists of 9 sets of time series. Each set holds 6 images of one plant taken over time (day 1, day 4, day 8, day 12, day 16 and day 20 of the growth period). Of these 54 images, 34 images (day 4 -day 20) are analysed, the day 1 images (as well as two day 4 images) are too early in the growth process and therefore too small in structure to be represented by a non-trivial Reeb graph.

## 4.1 Image Acquisition

At the GMI the seeds are planted on a nutrient containing agar gel surface [42] in plastic petri dishes (plates). One of these petri dishes holds 24 plants (2 rows of 12 plants each). Figure 4.1 shows one of these petri dishes, the plants in this image are already 20 days old. The plates are stored vertically in growth chambers that allow for controlled conditions such as constant temperature, illumination or humidity. The vertical orientation of the plates is based on the fact that roots grow primarily in the direction of gravity. The amount of water, or for example certain nutrients, further affect the speed of the growth process of the roots.
The images are taken using an image scanner. A special fixture allows for two plates to be placed in an exact known position inside the scanner. A ruler at this fixture is imaged together with the plate and serves as a reference marker. This allows for later measurements on the image to be converted from pixel measurements to real world dimensions as for example millimetre.

The images are acquired with a scan at 1200 dpi resolution with 8bit colour depth, therefore one image is of approximately 6000x6000 pixels in size. The images are stored as bmp files of

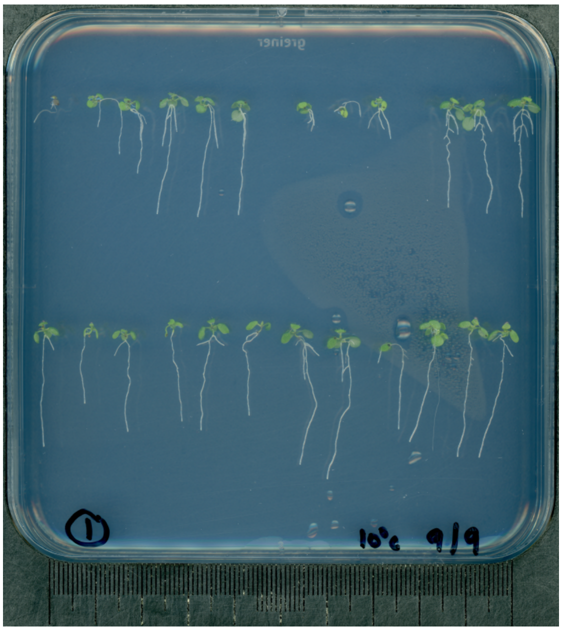
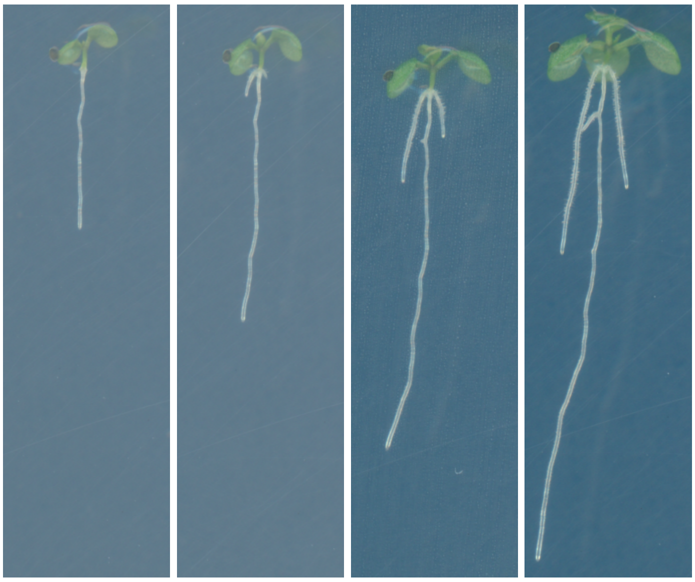


Figure 4.1: Plants grown inside a petri dish, picture on day 20 of their growth cycle. The ruler at the bottom of the image is used as a reference marker for measurements on the image.



(a) day 8 (b) day 12 (c) day 16 (d) day 20

Figure 4.2: Example images of the root dataset: root004 on several days during the growth cycle.

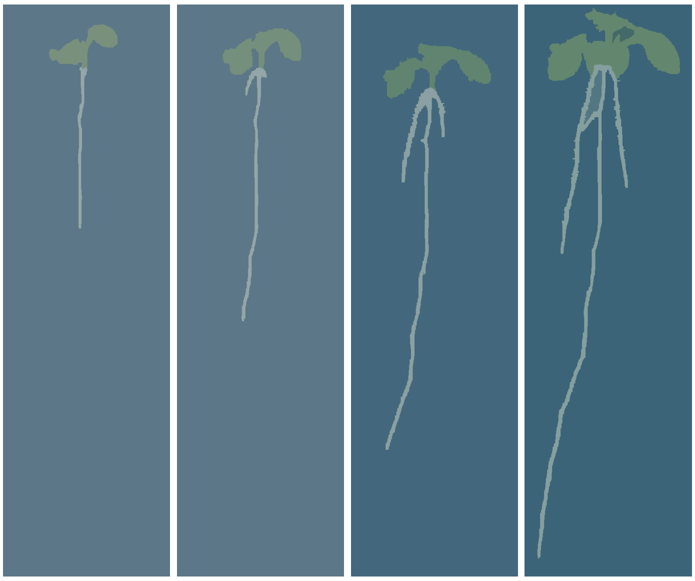
about 150MB.
Along time several successive images are acquired this way, as each plate is scanned at several successive days of the growth process. A 3D stack of 2D images over time is thus created for each root. An example for such successive images of one plant is shown in Figure 4.2

## 4.2 Pre-Processing

The approaches presented in this thesis take as input image a segmented image of a single plant. Therefore as a first step in the process of these graph based root representations pre-processing methods need to be applied.
As a set of 24 plants is pictured in one petri dish, each of these images needs to be cut to single plant images. The size of these single plant images is determined based on the image of the last day of the growth cycle. All younger plants are extracted accordingly. For the images in Figure 4.2 this pre-processing step has already been carried out.

The single plant images are segmented in the next pre-processing step. For this segmentation procedure a pyramid based approach is used. Image pyramids describe an image at multiple levels of resolution. The bottom level of the pyramid corresponds to the high resolution of the original image. For higher levels the resolution decreases based on a reduction factor applied to the pixels within the reduction window [27]. The segmentation approach is presented in [17] and [16]. It takes an arbitrary image as input and returns a hierarchy of segmented images. In a next step the user can interactively modify the segmentation: The user has the possibility to merge regions or prevent them from being merged in higher levels of the hierarchical segmentation. For changes, made by the user, the pyramid is recomputed to adapt the underlying combinatorial maps [17]. All images of the root dataset used in this thesis were segmented using this interactive, hierarchical segmentation tool.

The segmented images of the root dataset consist of 2 foreground regions (leaves and roots, only the roots are analysed for this approach) and up to 2 holes in the foreground structure. Due to this required segmentation, the dataset is restricted in its size, as this semi-automatic segmentation approach is costly in terms of time. For the thin root structures a high amount of user interaction is needed to generate a suitable segmentation result. Figure 4.3 shows the segmentation results for the root images shown in Figure 4.2.
More details on the complete root dataset are shown in [24].



(a) day 8 (b) day 12 (c) day 16 (d) day 20

Figure 4.3: Segmented images of root004, day 8 to day 20.

# CHAPTER 5

# Graphs on Root Images

Within the scope of this thesis, root images are represented by three alternative graph based representations: graphs based on a medial axis skeleton, Reeb graphs based on the height function as Morse function and Reeb graphs based on the geodesic distance as Morse function.

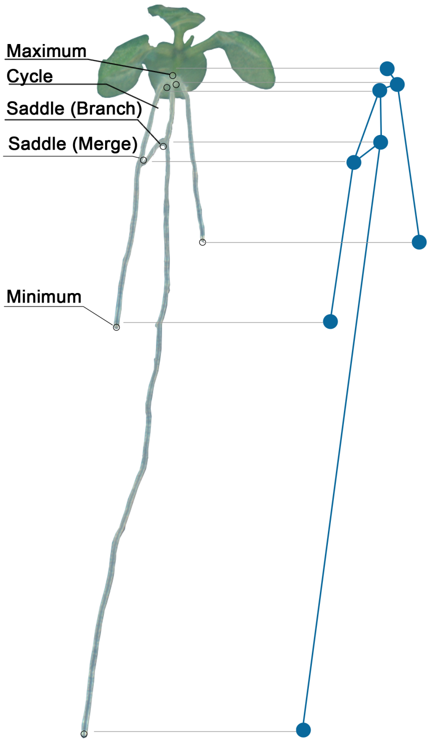


Figure 5.1: Example root with characteristic locations labelled and a possible graph representation.

These graph representations capture characteristics of the roots such as branching points and end points of branches. Figure 5.1 shows an example image of a root, with the characteristic points labelled.

An overview of the implementation is given as pipeline for the three approaches to achieve the graph based representations presented in this thesis in Figure 5.2. The approaches are discussed in detail in the following sections.

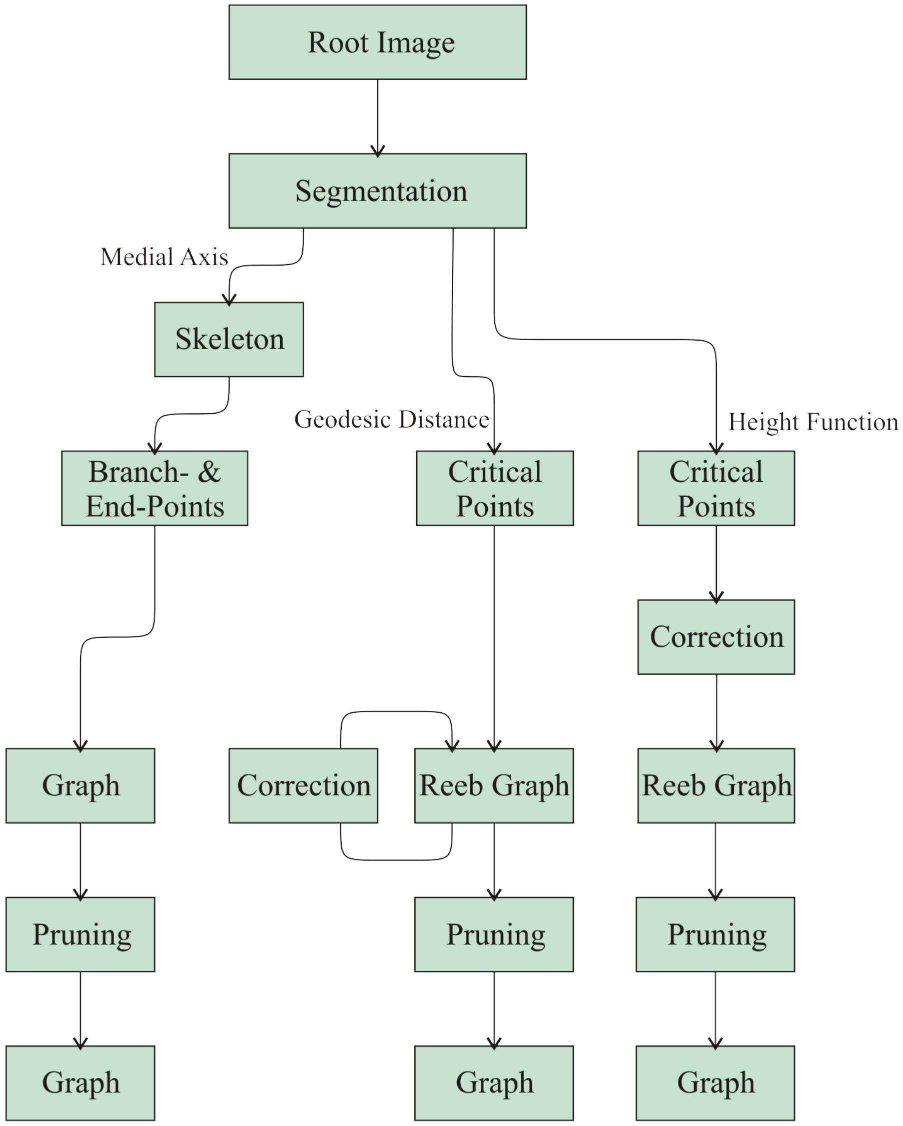
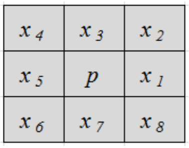


Figure 5.2: Implementation pipeline for the three approaches.

## 5.1 Computation of Medial Axis Graph

For the computation of the medial axis an iterative thinning approach is applied to the foreground region of the segmented image. Such a thinning procedure generates a one pixel thick representation of the root. However this only approximates the medial axis as an exact reconstruction is not possible for structures of even width [43].
For each pixel, in order to determine whether the pixel is a skeleton pixel, three conditions ($G_1$, $G_2$ and $G_3$, respectively $G_3'$ ) based on the neighbourhood of the pixel are checked in two sub-iterations. These two sub-iterations make one iteration of the thinning algorithm. If all three conditions hold for one sub-iteration, the pixel is removed. If one or more of the three conditions do not hold, the pixel is identified as a skeleton pixel and kept. The thinning algorithm is set to be repeated for an infinite number of iterations, but stops when the foreground region cannot be thinned further. For each skeleton pixel identified in this way, the shortest distance to the background is additionally stored as the width of the root.
For the thinning the conditions are checked on the neighbourhood of a pixel $p$. The east-neighbour is labelled $x_1$, the others are labelled with successive numbers counter-clockwise as illustrated by the following Figure 5.3 (pixel $x_9$ is defined as being equal to $x_1$):



| | | |
|---|---|---|
| $x_4$ | $x_3$ | $x_2$ |
| $x_5$ | $p$ | $x_1$ |
| $x_6$ | $x_7$ | $x_8$ |

Figure 5.3: Pixel neighbourhood.

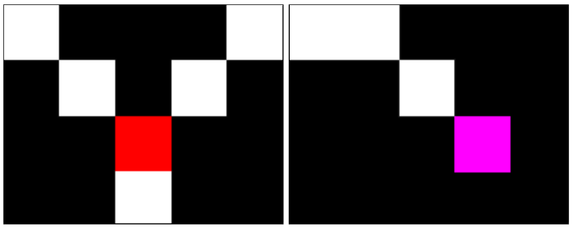
According to [28] the following conditions are checked for pixel $p$:

1. sub-iteration: the pixel $p$ is deleted if the conditions $G_1$, $G_2$ and $G_3$ hold:

    - $G_1$: $\sum_{i=1}^{4} b_i = 1$, with:
      $$b_i = \begin{cases} 1 & \text{if } x_{2i-1} = 0 \text{ and } (x_{2i} = 1 \text{ or } x_{2i+1} = 1), \\ 0 & \text{otherwise.} \end{cases}$$
    - $G_2$: $2 \leq min(n_1(p), n_2(p)) \leq 3$, with:
      $n_1(p) = \sum_{k=1}^{4} x_{2k-1} \vee x_{2k}$ and $n_2(p) = \sum_{k=1}^{4} x_{2k} \vee x_{2k+1}$
    - $G_3$: $(x_2 \vee x_3 \vee \neg x_8) \wedge x_1 = 0$

2. sub-iteration: the pixel $p$ is deleted if the conditions $G_1$, $G_2$ and $G_3'$ hold:

    - $G_1$: same as for sub-iteration 1
    - $G_2$: same as for sub-iteration 1
    - $G_3'$: $(x_6 \vee x_7 \vee \neg x_4) \wedge x_5 = 0$



(a) branching point (b) endpoint

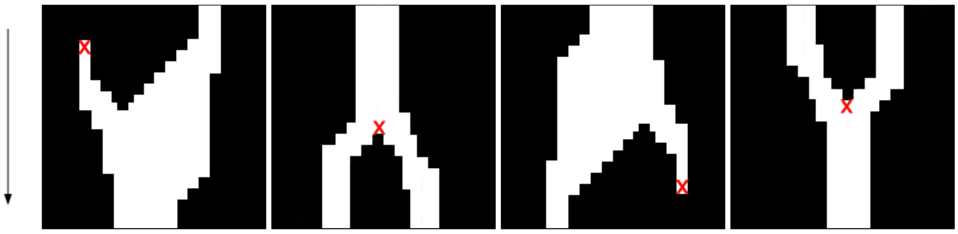
Figure 5.4: Branching point in a medial axis skeleton marked red, endpoint marked pink.

For each computed skeleton, branching points and endpoints of branches are detected according to their neighbourhood-structure. These points are used as nodes of the graph. An endpoint is detected as a pixel with only one neighbouring skeleton pixel (see Figure 5.4b), a branching point has more than two neighbouring skeleton pixels (see Figure 5.4a).
Finally the actual graph is built on the skeleton using line tracing and stored as an adjacency matrix. For each branching point all skeleton connections are followed until another branching point or an endpoint is reached. If a branching point or endpoint is reached, the corresponding entry in the adjacency matrix is set to one.

## 5.2 Computation of Reeb Graphs with the Height Function

For the computation of the critical points of the foreground region according to the height function, the basic idea is to analyse the borders of the foreground region in the root image. The borders of the foreground region (defined as the root region in the segmentation image) are therefore analysed with regard to horizontal borders (borders parallel to the x-axis) as these might describe a change in the number of components.

**Proposition 1** *Critical points according to the height function as given in Definition 16*[1]*, only occur at the border of a region but not within a region.*



(a) maximum / birth (b) saddle (split) (c) minimum / death (d) saddle (merge)

Figure 5.5: Four different types of critical points, computed according to the height function [25].

[1] Definition 16 on page 13

**Proof 1** *The function values computed on the foreground region are strictly monotonically increasing, when analysing the foreground in the direction of the height function. According to Definition 10* [2]*, critical points occur at the positions of local maxima or minima. Therefore the critical points according to the height function can only appear at the border of a foreground region.*

Based on this proof, critical points for the height function appear with horizontal borders (perpendicular to the direction of the height function) in the image. Therefore each horizontal border is followed until a vertical border is reached at its ends. As critical points only occur for local maxima and minima, the connected vertical borders are checked:

- no change in topology - no critical point: one end of the border connected to a vertical border that is in the direction of increasing height values, the other connected to a vertical border that is in the direction of decreasing height values
- split or birth: both endpoints connected to a vertical border that is in the direction of increasing height values (local minimum)
- merge or death: both endpoints connected to a vertical border that is in the direction of decreasing height values (local maximum)

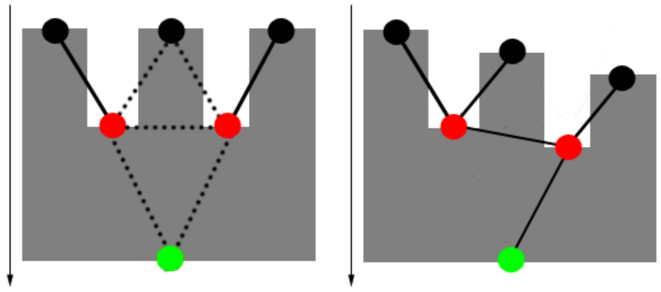
The so found critical points are located at the centre of the horizontal edge. Figure 5.5 shows examples for computed critical points.

### Correction of critical points to satisfy the definition of Morse theory

Due to the resolution of the image, the discretisation of the root and further distortions during the segmentation process, two critical points may be computed at the same height. Critical points with the same Morse function value contradict condition $M_3$ in Definition 11 [3] of Morse functions. For critical points on the same height a unique Reeb graph cannot be built, as the connections of the nodes in the graph are ambiguous [25]. Figure 5.6a illustrates the problem. In order to solve ambiguous edges of critical points on the same height, the following technique is used:
For two critical points on the same height, a correction is applied by adding a factor $f$, $0 \leq f \leq 1$. A critical point $p = (x, y)$ is translated to $p\prime = (x, y + f)$ with $f$ computed as $f = \frac{1}{w} \cdot (x - 1)$, $w$ being the image width. The order of heights is preserved by this controlled shift. The critical points are mathematically discriminative after applying this correction, but stay in the original image row, when rounding down the $y$-coordinate of such a critical point to an integer value [25]. The configuration of the nodes in Figure 5.6a is moved to the configuration in 5.6b. As the critical points are now on different heights, a unique Reeb graph can be built. The Reeb graph for Figure 5.6a after application of the proposed correction by factor $f$ is drawn in Figure 5.6b.

[2] Definition 10 on page 11

[3] Definition 11 on page 12



(a) critical points on same height (b) corrected critical points

Figure 5.6: For critical points on the same height a unique Reeb graph cannot be built: solid lines indicate well-defined connections in the graph, the dotted lines indicate possible connections. From these possible connections, one connection from the black centre node to a red node is needed and the green node needs to be connected to one of the red nodes as well.

### Construction of the graph

The identified critical points of the foreground region are represented by nodes in the Reeb graph.

**Proposition 2** *These nodes are level-set curves according to Definition 14* [4].

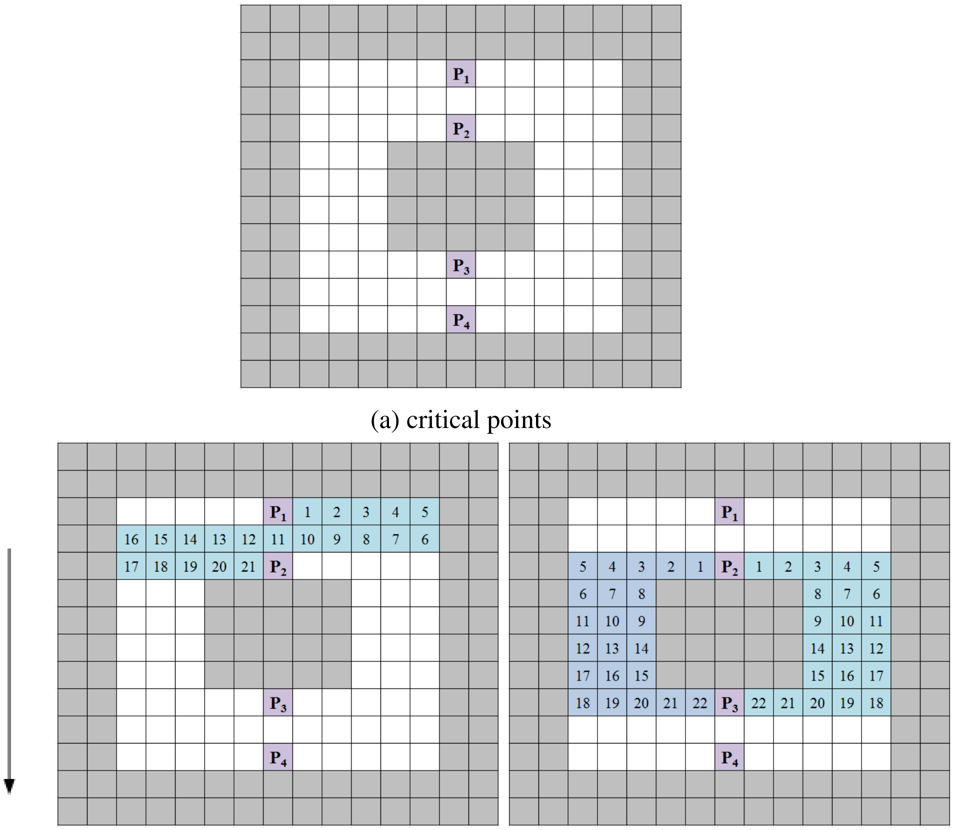
**Proof 2** *Each node represents a set of points (pixels) that form a connected component in an image row. These points have the same Morse function value according to the height function. Furthermore, for each point in the point set there is a path to at least one other point in the set. Hence, the points form a so-called connective point set.*

As a final step the edges of the graph need to be inserted. The edges are stored as an adjacency matrix.

In order to obtain the connections between nodes, all critical points are sorted according to their height. Starting with the topmost critical point an approach similar to the flood fill algorithm is deployed for all nodes:

1. The image row of the current critical point is followed to the right until a critical point in the same image row or a background pixel is reached.
2. In case a background pixel was reached, the search is moved one row down and this image row is followed to the left, until a critical point or a background pixel is reached.
3. In case a background pixel was reached, the search again moves down one row and progresses to the right until a critical point or background pixel is reached.

[4] Definition 14 on page 13



(a) critical points

(b) edge $P_1P_2$

(c) cycle $P_2P_3$

Figure 5.7: Computation of edges in a Reeb graph

4. Step 2 and 3 are repeated until the adjacent critical point is found.

Once the search terminates, a pair of adjacent critical points is identified and the adjacency matrix is set to one for the according edge. In case the search started from a critical point of type saddle (split) the search for critical points is done twice, once starting to the right of the critical point and once to the left. Figure 5.7 shows how the connections are computed for the critical points $P_1$ (Figure 5.7b) and $P_2$ (Figure 5.7c) in the white foreground region of Figure 5.7a.

**Proposition 3** *The edges resulting from the flood-fill approach described above connect adjacent level-set curves.*

**Proof 3** *According to the Definitions 15* [5] *and 12* [6]*, level-set curves are adjacent if their point sets are connective. A flood fill algorithm only reaches points in a connective point set. A critical point $p_2$ with height function value $h_2$ is reachable by a flood fill algorithm from a critical point*

[5] Definition 15 on page 13

[6] Definition 12 on page 13

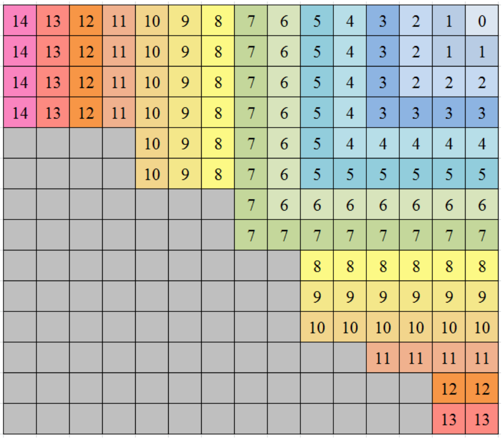
*$p_1$ with height function value $h_1$ ($h_1 < h_2$). The pixels $p_i$ at function values $h_1 \leq h_i \leq h_2$, covered by the flood fill algorithm, form a connective point set, therefore the level-set curves are adjacent.*

## 5.3 Computation of Reeb Graphs with the Geodesic Distance

For each foreground pixel the geodesic distance to one predefined source pixel is computed. For the root images the distance is computed from the centre pixel of the topmost pixel row. As a distance measurement for computing the critical points the Chebyshev (chessboard) distance is used. Starting from one pixel, all pixels in the 8-neighbourhood have distance one to the source pixel. The distance therefore spreads in concentric squares. Based on the discrete pixel grid of a 2D image, this allows for an easy computation of critical points. An approach similar to the one used for the height function in Section 5.2 can be employed: The foreground (defined as the root region in the segmented image) is analysed, looking for a change in the number of connected components (topology changes) from one distance value to the next. Critical points can be identified in this way.
A maximum node describes the birth of a new connected component, a minimum node the death of a connected component.

**Proposition 4** *There is only one maximum node in a Reeb graph, built based on the geodesic distance, which is the source point.*

**Proof 4** *As the distance values for all foreground pixels are computed starting from the source pixel, the distance increases with the distance to the source point. Therefore there is only one maximum node at distance 0 (at the source point).*



| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 1 |
| 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 2 | 2 |
| 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 3 | 3 | 3 |
| | | | | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 4 | 4 | 4 | 4 |
| | | | | 10 | 9 | 8 | 7 | 6 | 5 | 5 | 5 | 5 | 5 | 5 |
| | | | | | | | 7 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| | | | | | | | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| | | | | | | | | | 8 | 8 | 8 | 8 | 8 | 8 |
| | | | | | | | | | 9 | 9 | 9 | 9 | 9 | 9 |
| | | | | | | | | | 10 | 10 | 10 | 10 | 10 | 10 |
| | | | | | | | | | | | 11 | 11 | 11 | 11 |
| | | | | | | | | | | | | | 12 | 12 |
| | | | | | | | | | | | | | 13 | 13 |

Figure 5.8: Geodesic distances are computed for this thesis using the chessboard distance. Here the source pixel is located in the top right corner. At distance 8 the foreground is split into two connected components, this would result in a critical point of type saddle (split) at distance 7.

**Proposition 5** *Minimum nodes are found at the position of a maximum distance in a connected component (local maximum).*

**Proof 5** *As the source point for the geodesic distance is located at the top of the root and the distance is strictly monotonically increasing inside the root, a connected component (a branch of the root) can only end in a local maximum.*

**Proposition 6** *Saddle points are determined as locations at which foreground regions, with the same distance to the source, are split into two connected components or are merged from two into one connected component.*

**Proof 6** *Nodes in the Reeb graph represent level-set curves (see Definition 14* [7]*). For each connected set of points (pixels) at a certain distance a node may represent this level-set curve. According to Definition 19* [8] *a node corresponding to a critical point of type saddle has degree three. In case a connective point set at the next smaller or larger distance (distance-1 or distance+1) exists for a point set, the level-set curves of these point sets are adjacent. Therefore, the level-set curve of a saddle node is adjacent to three other level-set curves. Two of these level-set curves need to be at the same distance but represent non-connective point sets. Hence, the foreground is split into two components for this distance.*

Figure 5.8 shows an example for a critical point of type split. Here the source pixel is located in the top right corner (distance 0). At distance 8 the structure is split into two connected components, this would result in a critical point of type split at distance 7. The split in Figure 5.8 would not be determined as a critical point when analysing the structure according to the height function, as the number of connected components stays one for all heights in this structure. The geodesic distance is more flexible in the identification of critical points than the height function; an alignment of the analysed structure to the perpendicular axis is not needed.

### Correction of critical points to satisfy the definition of Morse theory

As for the Reeb graphs based on the height function, critical points with the same Morse function value (in this case distance to a source pixel) may occur also for Reeb graphs based on the geodesic distance. This contradicts condition $M_3$ in the definition of Morse functions (Definition 11 [9]). However, for this type of Reeb graph there is no direct correction applied to the critical points. The critical points are rather connected as described in the next section, to the next critical point that is encountered while tracing back the foreground structure from one critical point towards the source pixel. However, if two critical points at the same chessboard distance are encountered - this might for example happen for the closest points to the maximum node - a second distance measurement is used for the decision. Starting from the source pixel, the geodesic distance is computed based on the Euclidean distance.

The values obtained by the Euclidean distance are never smaller than the chessboard distance

[7] Definition 14 on page13

[8] Definition 19 on page 14

[9] Definition 11 on page 12

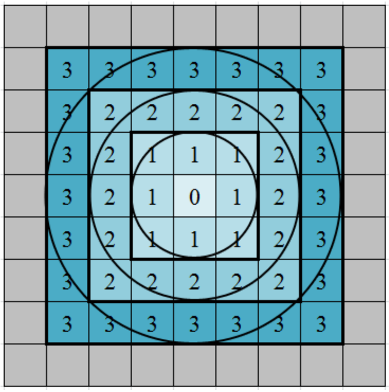


Figure 5.9: Geodesic distance computed as chessboard distance from the centre pixel leads to concentric squares. The Euclidean distance is described by the concentric circles.

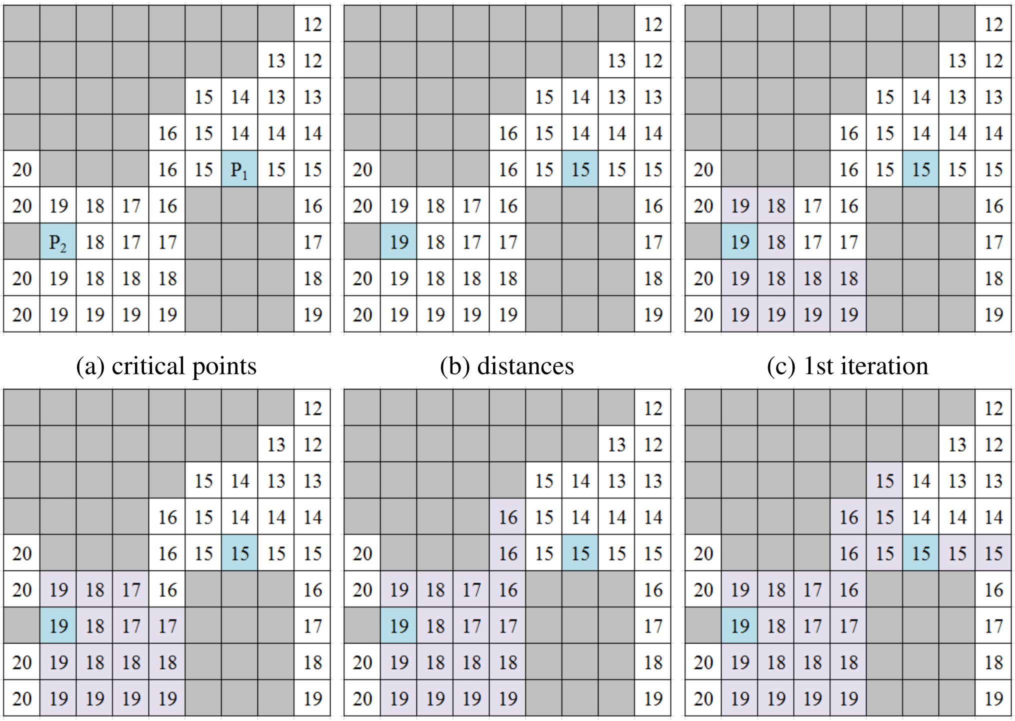
as the Euclidean distance is the radius of the incircle of the square formed by the chessboard distance (see Figure 5.9).
The risk of encountering two critical points in the same branch having the same Euclidean geodesic distance to a source pixel is neglected within the scope of this thesis: For the discrete pixels of an image, two pixels may have the same Euclidean distance to a source pixel. However, with increasing distance to a source pixel the variance in the Euclidean distance values increases as well. All pixels at a chessboard distance $d$ are represented by $d + 1$ different Euclidean distances. Moreover, the thin root structures decrease the possibility of encountering two pixels with the same Euclidean distance to a source pixel in the same branch.
Using the Euclidean distance as decision criteria may result in wrong decisions. One saddle node may be adjacent to four other nodes, while another saddle node only has two adjacent nodes. As illustrated in the “geodesic distance” branch in Figure 5.2 a correction procedure is applied to repair the graph in case of incorrect decisions.

### Construction of graph

Based on the computed critical points, that form the nodes of the Reeb graph, the edges in the graph are obtained in two steps (a correction may be needed as a third step):

1. First all adjacent critical points of type saddle or merge are connected. For each saddle node the foreground is traced backwards in the direction of decreasing distances until a critical point is reached. This means starting at a saddle node $P$ with distance $x$ to the source node, the algorithm moves from the region with distance $x - 1$ that is reachable from $P$ to $x-2$ etc. until a critical point is found. Figure 5.10 demonstrates this procedure for the critical point $P_2$ at distance 19. For saddle nodes of type merge this is done for both branches until two connections are found.

   **Proposition 7** *For each saddle node in the graph, there is always another adjacent node at a smaller distance.*



(a) critical points (b) distances (c) 1st iteration

(d) 2nd iteration (e) 3rd iteration (f) critical point found

Figure 5.10: Example for the computation of edges in the geodesic distance Reeb graph. The connection between critical point $P_2$ and $P_1$ is built.

**Proof 7** *For each saddle node in the graph there exists another node at a smaller distance to the source pixel, which is either a saddle or a maximum node. The source pixel is at distance 0 and forms a maximum node in the graph. Hence, the source pixel is always at a smaller distance than any saddle node in the graph.*

2. As the second step, the maximum node (source node) in the Reeb graph is connected to the node closest to it. All minimum nodes are connected to the closest split or merge node. Again the foreground is traced backwards in the direction of decreasing distance until a critical point is found.

**Proposition 8** *For the maximum node, there is always an adjacent saddle or minimum node in the graph. For each minimum node in the graph, there is always another adjacent node at a smaller distance.*

**Proof 8** *Each Reeb graph contains at least two nodes, as there is always a maximum node representing the birth of component and a minimum node representing the death of said component. Therefore there is always at least a minimum node adjacent to the maximum* *node. For a branched structure, further saddle nodes are introduced in the graph according to Proof 6* [10]*. Therefore there exists a saddle node adjacent to the maximum node. This argument applies analogous for minimum nodes: there is at least the maximum node adjacent to the minimum node if there are only two nodes in the graph. Otherwise there exists an adjacent saddle node in the graph.*

3. For two critical points at the same distance, wrong decisions may be taken during the second step of this process. Therefore the correctness of the graph is checked after both steps are finished. In case a saddle node is adjacent to four other nodes, there needs to be a second saddle node at the same distance with only two adjacent nodes. For a saddle node with four adjacent nodes, the minimum or maximum node, that is at the smallest distance from a saddle node with degree two, is deleted from the adjacency matrix and connected to this saddle node that so far only had two adjacent nodes.

   **Proposition 9** *For saddle nodes with a degree higher than three, there is always a second saddle node at the same distance with a degree smaller than three, so that the correction presented above may be applied.*

   **Proof 9** *Based on Definition 19* [11] *a saddle node in a Reeb graph has degree three. For minimum and maximum nodes only one edge is computed. These nodes will always have degree one. Therefore a saddle node of degree >3 always causes a saddle node of degree<3.*

All edges that are computed in that way are stored in an adjacency matrix.

Again the nodes of the Reeb graph are level-set curves and the edges connecting nodes connect adjacent level-set curves. Proof 2 [12] and 3 [13] provided for the Reeb graphs based on the height function apply for Reeb graphs based on the geodesic distance as well. Based on the geodesic distance the nodes form level-set curves for a set of points with the same distance to a source point. The computation of the edges in the graph is again based on an algorithm that traces the foreground. For the geodesic distance the foreground is not traced in succeeding image rows according to the height function but following succeeding distance values and their level-set curves in the foreground.

## 5.4 Improvements on the Graphs

During pre-processing the image is segmented. This segmentation procedure may generate frayed borders in the foreground region. These artefacts introduce additional, spurious nodes in the graphs. In order to use the graphs as a representation of the root structure, these additional

[10] Proof 6 on page 37

[11] Definition 19 on page 14

[12] Proof 2 on page 34

[13] Proof 3 on page 35

branches need to be removed. A simple graph pruning approach as for example described in [3] is applied for this purpose. Branches and endpoints are discarded according to certain criteria. If needed, nodes are relinked after pruning. Saddle nodes that are reduced to a degree-2 node by the pruning process are smoothed out (see Definition 5 [14]) and therefore disappear in the graph. The graphs representing the root dataset according to the implementation described in the Sections 5.1, 5.2 and 5.3 are connected graphs according to Definition 4. The pruning procedure described in the following sections does not change to connectedness of the graphs. However, for a different dataset, that might contain disconnected graph representations (for example due to segmentation artefacts) these disconnections need to be corrected as well.

### Pruning based on length

This pruning approach identifies and discards edges that are smaller than a predefined length. The images are cut in a pre-processing step to single plant images. The size of the image for each plant is based on the latest day of its growth cycle. All earlier images of this plant are cut to the same size. The image height therefore is directly linked to the length of the longest branch as this determines the image size. The approach used within this thesis dismisses branches with an Euclidean distance between the endpoints of less than 1.5% of the image height. The threshold of 1.5% of the image height was determined empirically. For justification of this pruning threshold see the results in Chapter 6, Table 6.1.

### Pruning based on width

For the medial axis graph a second pruning approach based on the width of branches was tested. The basic idea is that branches due to segmentation artefacts are in most cases only one or two pixels wide. True branches that resemble the spurious branches in length are mostly broader than that. However, the pruning process cannot be based on the width of branches only, as especially longer branches tend to be thin in the end part and may be discarded during pruning.

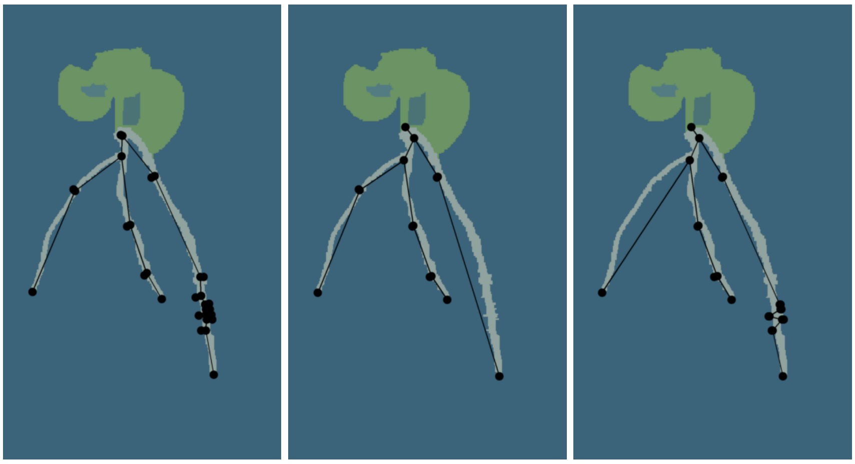
Therefore, pruning candidates are detected using the length based pruning approach described before. These pruning candidates are verified for pruning by checking the average width of the branch between the endpoints of the edge forming the branch. This can be done easily as the width of the foreground structure is stored with the medial axis skeleton. In case the average width is below seven pixels, the pruning candidate is verified as an artefact and the according branch is discarded. The width threshold again was determined empirically.

[14] Definition 5 on page 7

CHAPTER 6

# Results

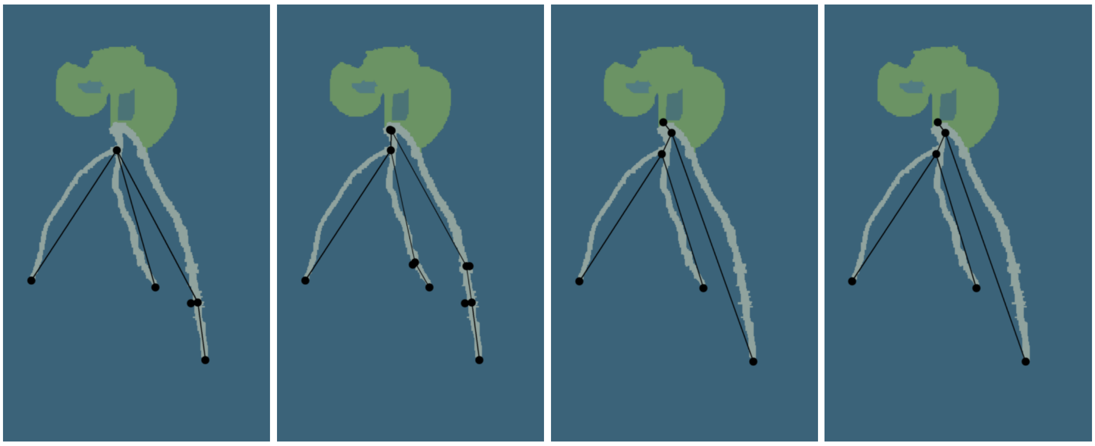
The overall goal presented in this thesis, is to define graph based representations of root images, that correctly capture the structure (branching points and endpoints) of the root as well as characteristics such as for example the length or width of branches.
Furthermore, the different graph based representations are compared. In order to combine characteristics captured by different representations, the equality of different representations for the same image needs to be evaluated.
For the results presented in the following sections, the root dataset of 34 root images (as defined in Chapter 4) was evaluated.
Figure 6.1 shows results of the graph representations for root 09, day 20. In this example the Reeb graph based on the height function produces the best result: the true branching points and



(a) medial axis graph (b) Reeb graph (height function) (c) Reeb graph (geodesic distance)

Figure 6.1: Resulting graphs for root 09, day 20.

endpoints of branches are correctly detected and the representation introduces the least spurious branches. This is due to the fact that the root is well aligned with the perpendicular axis and the response to the artefacts introduced by the segmentation approach is lower for this type of Reeb graph compared to the Reeb graph based on the geodesic distance. This behaviour is based on the restriction of the critical points of the height function, as these only occur with horizontal borders of gaps in the foreground structure. The medial axis responds very sensitive to noise in the foreground borders, the high number of spurious branches shows this.
Figure 6.2 shows again the resulting images for root 09, day 20 with length based graph pruning applied. By application of graph pruning all spurious branches are detected and discarded for the two Reeb graph representation. The resulting graphs based on the Reeb graph approaches after pruning therefore capture the characteristics, that were intended to be obtained, well: startpoint and the three endpoints of the roots are found, as well as the two branching points. For the medial axis representations under pruning, the number of spurious branches was reduced by up to 90% (depending on the pruning approach). The actual startpoint (for the medial axis there is no special node of type maximum) is not secure under pruning and was discarded in the pruning approach. This created a regular node (degree-2) for the higher branching point in which thus was discarded as well. While the startpoint and one branching point are missing, one spurious branch was kept. By application of width based pruning described in Section 5.4, the missing startpoint and branching point are kept, but two more spurious branches (in total three spurious branches) are kept just as well (see Figure 6.2b.)



(a) medial axis graph (length pruning)

(b) medial axis graph (width pruning)

(c) Reeb graph (height function)

(d) Reeb graph (geodesic distance)

Figure 6.2: Pruned graphs for root 09, day 20 - length based graph pruning was applied to the results shown in Figure 6.1. For the medial axis based representation the width pruning approach was applied as well.

## 6.1 Pruning Effect

For the comparison of the three different graph representations used in this thesis, the numbers of nodes computed for the graphs are compared. As illustrated by the Figures 6.1 and 6.2 spurious branches may be introduced due to segmentation artefacts. Due to these segmentation artefacts the number of nodes is in general high for all three representations, but can be reduced by up to 70% under the application of graph pruning (see Table 6.2). The applied graph pruning is able to remove most of the spurious branches. However, spurious branches may still be accepted using graph pruning (false positives), while true branches that resemble the spurious branches (in length) are therefore discarded during the pruning process (false negatives). Table 6.1 shows the number of false positives and false negatives that occur for the root dataset of 34 images for each of the discussed approaches. More than one false positive or false negative or even both incorrect decisions may occur in one image. Thus, the number of images in the root dataset for which incorrect decisions were taken is given in the column "#images" as well. The medial axis approach introduces the most false positives, especially the width pruning approach is very likely to accept spurious branches. However, the number of false negatives here is lower than for the medial axis with graph pruning based on length only.

Table 6.2 shows the numbers of nodes for all root images in the dataset and the reduction of the node numbers when using graph pruning. The number of nodes in the Reeb graphs without graph pruning is high compared to the human ground truth, for all three representations. This is due to artefacts in the segmentation images, that create spurious branches in the graphs. For the images of the roots 04 and 05 on day 16 the number of nodes in the Reeb graphs is higher than the number of nodes in the Reeb graphs of these roots on day 20. The images of day 16 show a lot of noise due to humidity (fog and water drops on the surface of the petri dishes) which created artefacts in the segmentation images and thus spurious branches in the graphs.
Graphs that contain a cycle are marked with a $\star$ sign, respectively $\star\star$ for two cycles in the graph. The number of edges is not mentioned in the table as it is implicitly given through the number of nodes and cycles in the graphs according to Definition 3 [1]. The graphs generated for the root dataset were compared to human generated ground truth as well. Graph representations that are equal to the human ground truth in number of nodes as well as in structure are highlighted in bold font in Table 6.2.

Table 6.1: Branches wrongly discarded (false negative) and wrongly accepted (false positive) in the graph pruning approaches.

| **Wrong decisions on graph pruning** | | | |
|---|---|---|---|
| **representation** | **# images** | **# false negatives** | **# false positives** |
| RG height function | 11 | 9 | 5 |
| RG geodesic distance | 11 | 8 | 6 |
| medial axis | 21 | 18 | 11 |
| medial axis width pruning | 22 | 12 | 23 |

[1] Definition 3 on page 7

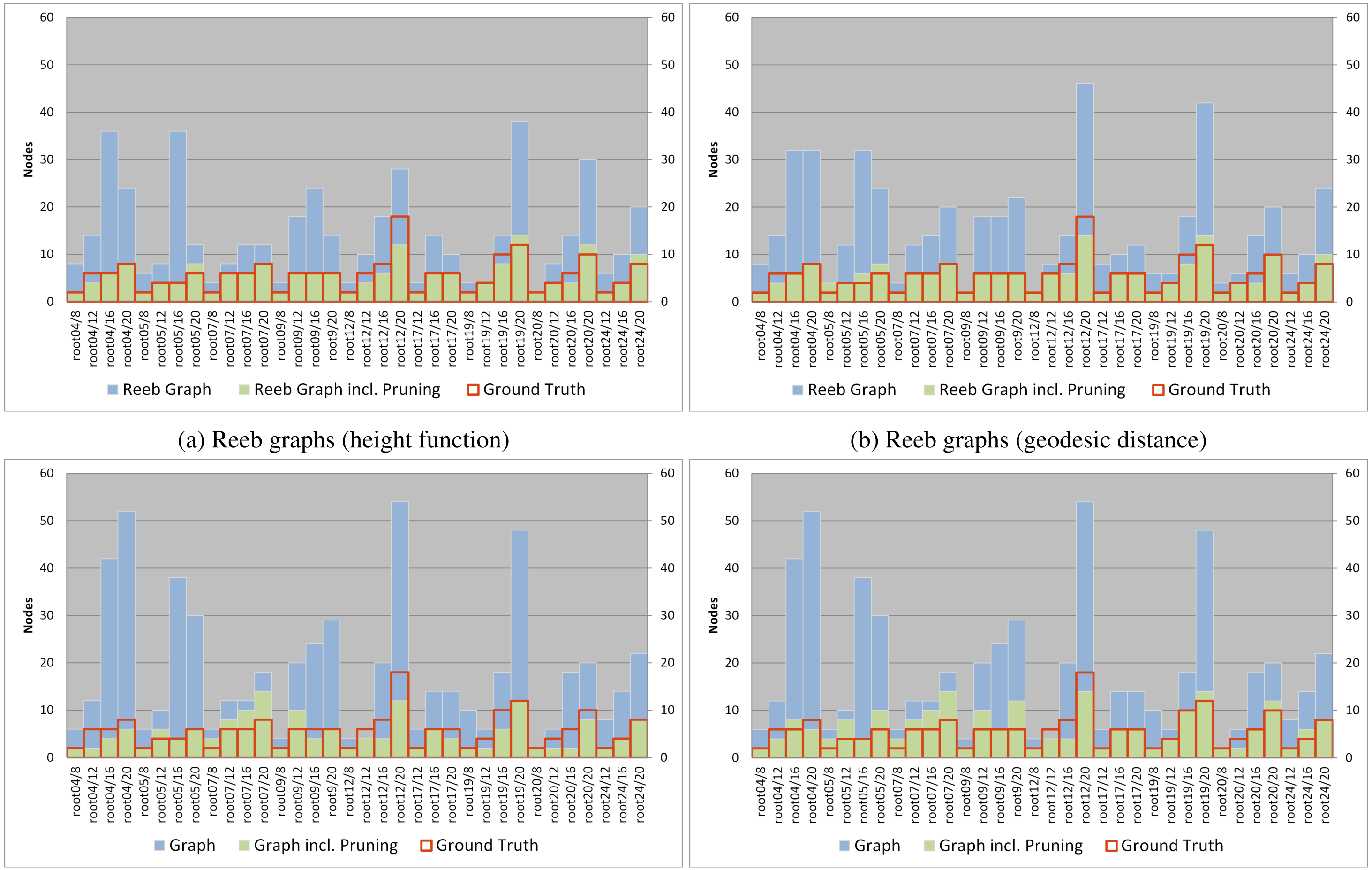
The abbreviations used in Table 6.2 are:

- RG1: Reeb graph based on height function
- RG1 + P: Reeb graph based on height function including pruning
- RG2: Reeb graph based on geodesic distance
- RG2 + P: Reeb graph based on geodesic distance including pruning
- MAG: Medial axis graph
- MAG + P: Medial axis graph including pruning
- MAG + WP: Medial axis graph including width based pruning
- RSA: Result from the Root System Analyser tool [31]
- GT: Human generated ground truth

Figure 6.3 shows the according graphs to the results presented in Table 6.2. The graphs show the numbers of nodes in the graph representations with and without graph pruning, as well as the human generated ground truth.

The geodesic distance based Reeb graphs with graph pruning approximate the human ground truth best regarding the number of nodes in the graphs. As well the structure of the graphs according to the human ground truth is best approximated by the Reeb graph representations with graph pruning (65% matches). The numbers of nodes highlighted in bold in Table 6.2 indicate a match of the structure of the graph representation with the human generated ground truth. Out of the 34 images in the root dataset 23 graphs are equal to the human ground truth for each of the two Reeb graph approaches.

For the two medial axis representation only 13 respectively 12 root representations match the human generated ground truth. For some roots the structure of the medial axis graph does not at all match the ground truth, as branches are destroyed in the graph pruning approach. As there is no distinct start node in the medial axis graph, the node that corresponds with the start node in the Reeb graph representation may be discarded in the graph pruning process. The branching node that results in a degree-2 node is smoothed out of the graph and the branches are connected contrary to the actual branching pattern (see Figure 6.2a for an example).



(a) Reeb graphs (height function)

(b) Reeb graphs (geodesic distance)

(c) Medial axis graphs (pruning = length pruning)

(d) Medial axis graphs (pruning = width pruning)

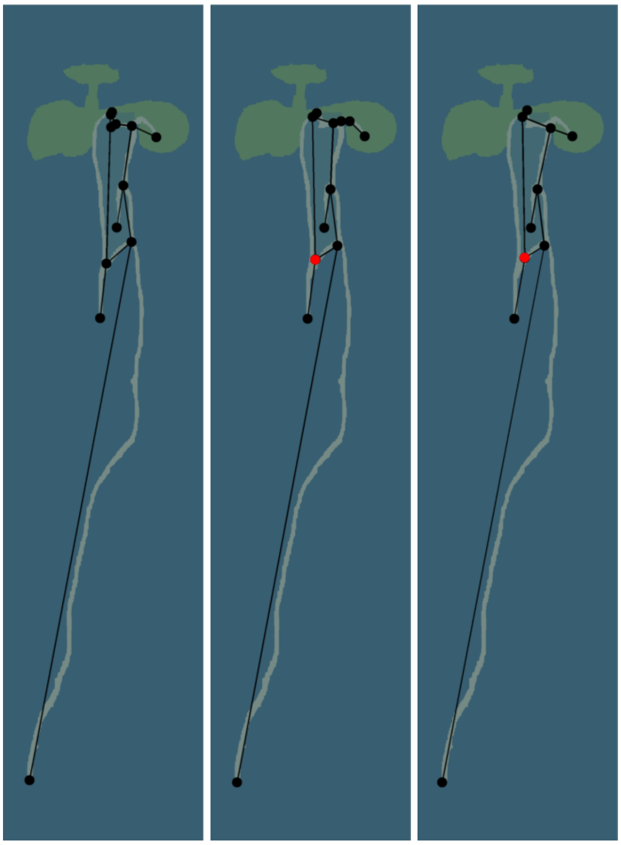
Figure 6.3: Numbers of nodes used in the graphs compared to human ground truth.

Table 6.2: Overview: number of nodes for the different graphs representations.

| **Number of nodes in the graph** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Image** | **RG1** | **RG1 + P** | **RG2** | **RG2 + P** | **MAG** | **MAG + P** | **MAG + WP** | **RSA** | **GT** |
| root04, day 8 | 8 | **2** | 8 | **2** | 6 | **2** | **2** | 5 | **2** |
| root04, day 12 | 14 | 4 | 14 | 4 | 12 | 2 | 4 | **6** | **6** |
| root04, day 16 | 36 | **6** | 32 | **6** | 42 | 4 | 8 | 25 | **6** |
| root04, day 20 ⋆ | 24 | **8** | 32 | **8** | 52 | 6 | 6 | 32 | **8** |
| root05, day 8 | 6 | **2** | 4 | 4 | 6 | **2** | 4 | **2** | **2** |
| root05, day 12 | 8 | **4** | 12 | **4** | 10 | 6 | 8 | 7 | **4** |
| root05, day 16 | 36 | **4** | 32 | 6 | 38 | **4** | **4** | 15 | **4** |
| root05, day 20 ⋆ | 12 | 8 | 24 | 8 | 30 | 6 | 10 | 21 | **6** |
| root07, day 8 | 4 | **2** | 4 | **2** | 6 | 4 | 4 | 6 | **2** |
| root07, day 12 | 8 | **6** | 12 | **6** | 12 | 8 | 8 | 11 | **6** |
| root07, day 16 ⋆ | 12 | **6** | 14 | **6** | 12 | 10 | 10 | 9 | **6** |
| root07, day 20 ⋆ | 12 | **8** | 20 | **8** | 18 | 14 | 14 | 18 | **8** |
| root09, day 8 | 4 | **2** | **2** | **2** | 4 | **2** | **2** | **2** | **2** |
| root09, day 12 | 18 | 6 | 18 | 6 | 20 | 10 | 10 | 17 | **6** |
| root09, day 16 | 24 | **6** | 18 | **6** | 24 | 4 | **6** | 16 | **6** |
| root09, day 20 | 14 | **6** | 22 | **6** | 29 | 6 | 12 | 24 | **6** |
| root12, day 8 | 4 | **2** | **2** | **2** | 4 | **2** | **2** | 3 | **2** |
| root12, day 12 | 10 | 4 | 8 | **6** | 4 | 4 | 4 | 4 | **6** |
| root12, day 16 | 18 | 6 | 14 | 6 | 20 | 4 | 4 | 14 | **8** |
| root12, day 20 ⋆ | 28 | 12 | 46 | 14 | 54 | 12 | 14 | 40 | **18** |
| root17, day 12 | 4 | **2** | 8 | **2** | 6 | **2** | **2** | 4 | **2** |
| root17, day 16 | 14 | **6** | 10 | **6** | 14 | **6** | **6** | 10 | **6** |
| root17, day 20 | 10 | **6** | 12 | **6** | 14 | 4 | 6 | 12 | **6** |
| root19, day 8 | 4 | **2** | 6 | **2** | 10 | **2** | **2** | 4 | **2** |
| root19, day 12 | **4** | **4** | 6 | **4** | 6 | 2 | **4** | **4** | **4** |
| root19, day 16 ⋆ | 14 | 8 | 18 | 8 | 18 | 6 | 10 | 12 | **10** |
| root19, day 20 ⋆⋆ | 38 | 14 | 42 | 14 | 48 | **12** | 14 | 28 | **12** |
| root20, day 8 | **2** | **2** | 4 | **2** | **2** | **2** | **2** | **2** | **2** |
| root20, day 12 | 8 | **4** | 6 | **4** | 6 | 2 | 2 | 3 | **4** |
| root20, day 16 | 14 | 4 | 14 | 4 | 18 | 2 | 6 | 13 | **6** |
| root20, day 20 ⋆ | 30 | 12 | 20 | **10** | 20 | 8 | 12 | 12 | **10** |
| root24, day 12 | 6 | **2** | 6 | **2** | 8 | **2** | **2** | 5 | **2** |
| root24, day 16 | 10 | **4** | 10 | **4** | 14 | **4** | 6 | 8 | **4** |
| root24, day 20 | 20 | 10 | 24 | 10 | 22 | **8** | **8** | 13 | **8** |
| **Sum** | **478** | **184** | **524** | **190** | **609** | **174** | **218** | **407** | **192** |
| **Reduction** | | **61.5%** | | **64.1%** | | **71.4%** | **64.2%** | | |

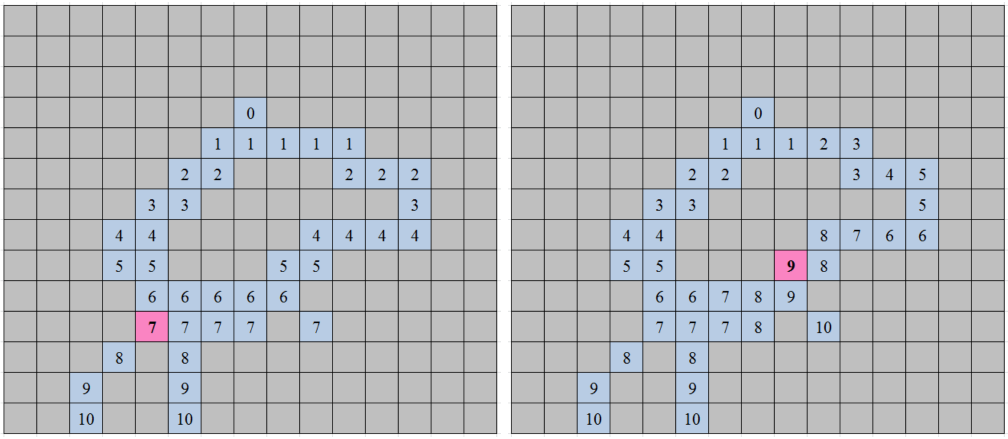
## 6.2 Overlapping Branches

The roots are imaged as a projection of a 3D structure (the root) to the 2D image space. Branches may therefore overlap in the 2D image. One major advantage when analysing roots based on Reeb graphs is posed by the ability to immediately distinguish between branching points and overlaps in the root structure [25]. An overlap introduces a cycle in the graph and therefore a saddle node of type merge in the set of critical points. For a medial axis based graph there are only two types of nodes: endpoints and branching points. Therefore, a merge of the foreground structure cannot be immediately distinguished from a split of the structure based on the type of node in the graph.



(a) medial axis graph (b) Reeb graph (height function) (c) Reeb graph (geodesic distance)

Figure 6.4: Resulting graphs for root 12, day 20: branches overlap in the image and form a cycle in the graphs. The saddle node of type merge is marked in red in the two Reeb graphs.



(a) merge node based on height function (b) merge node based on geodesic distance

Figure 6.5: Comparison of position of merge nodes in the two Reeb graphs.

Figure 6.4 shows an example of such an overlap of branches and the computed graphs for root 12, day 20. There is a cycle in all three images, however only in the Reeb graph representations the overlap is explicitly marked by the merge node (marked in red in Figure 6.4).

For Reeb graph representations based on the geodesic distance, the saddle node of type merge may not always be at or near the location of the actual overlap of branches as it is computed at the location of a merge of connected components with the same distance. As illustrated in Figure 6.5b this position can be inside one of the branches forming the cycle. For the Reeb graph based on the height function, the merge node is located as shown in Figure 6.5a. However, for the height function, saddle nodes of type merge may be introduced without any cycle in the graph. For the height function more than one maximum node may appear in the graph, these maximum nodes are connected to the foreground structure by a saddle node of type merge, without any cycle in the graph (respectively the hole in the foreground structure).

For the geodesic distance such branches are represented by split nodes. Merge nodes only appear for an overlap of the branches. However considering the merge node in Figure 6.5b the position of the merge node may be inside a branch instead at the actual overlap. This must be taken into account when solving the overlaps.

## 6.3 Comparison: Root System Analyser

The root dataset was evaluated using the tool “Root System Analyser” (RSA) [31] as well. This tool provides the number of branches for a root, the length of branches, as well as the width of a branch. The number of nodes computed by RSA is listed in Table 6.2 in column “RSA”. Although a smoothing is applied, a large number of nodes that represent noise is kept by this approach. This is indicated by the high number of nodes needed for the root dataset (compare: 407 nodes RSA to 190 nodes geodesic distance Reeb graph with pruning to 192 nodes human ground truth). Nodes of degree two that have no influence on the topology of the foreground

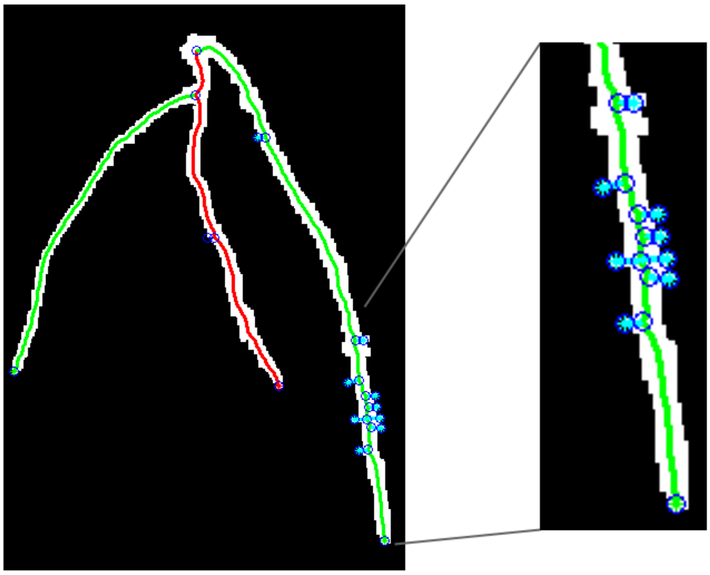


Figure 6.6: Result for root09, day 20. Computed using the “Root System Analyser”: Some small branches due to noise are kept in the final representation.

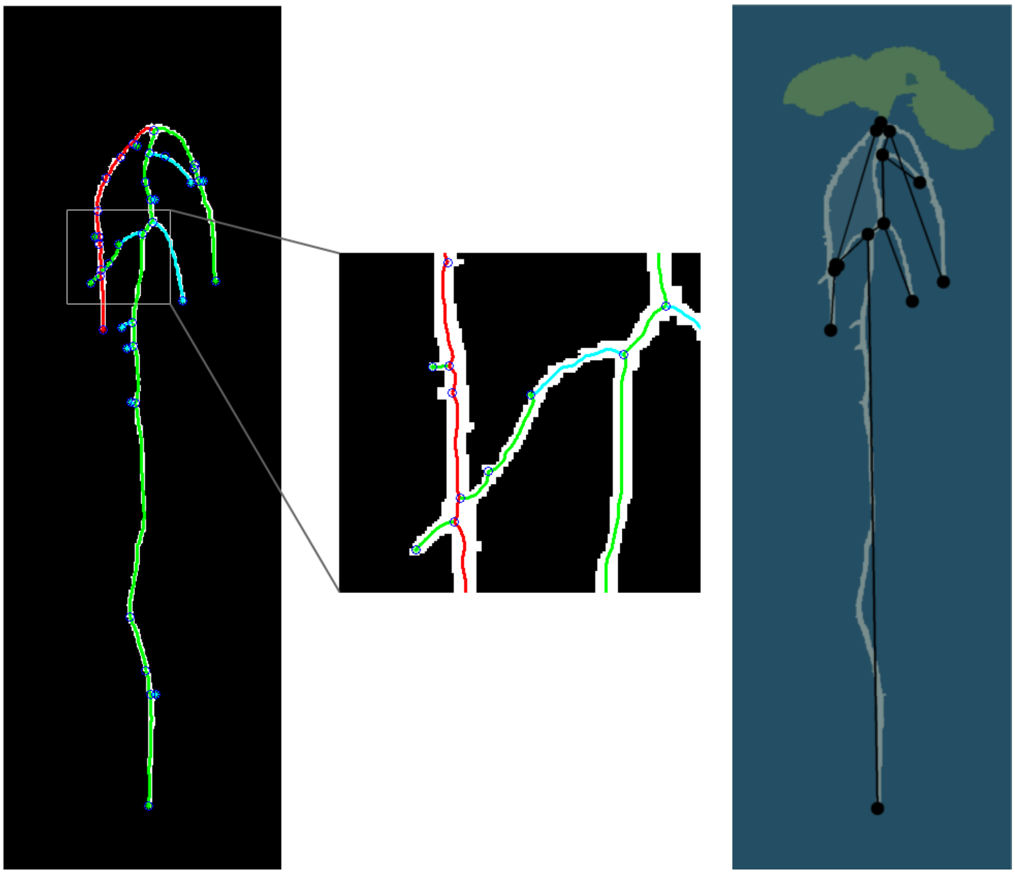
structure are kept in the representation as well.

### Spurious branches

The representation shown in Figure 6.6 was computed using RSA. The main branches of the root are well represented, some small branches due to noise are kept in the final representation. While both Reeb graph representations presented in this thesis use 6 nodes and 3 branches (see Figure 6.2) which conforms to the human ground truth, the RSA uses 23 nodes and detects 12 branches.

### Overlapping branches

Figure 6.7 shows a comparison of the representation computed by RSA and a representation computed using the geodesic distance based Reeb graph approach as presented in this thesis. The RSA representation (Figure 6.7a) detects 16 branches, using 40 nodes for the representation. The Reeb graph representation based on the geodesic distance with pruning (Figure 6.7b), needs 14 nodes for the representation and computes a total of 6 branches. However, two small branches are wrongly discarded during the graph pruning as is the end part of one of the overlapping branches. Just as the Reeb graph based representations, the RSA detects overlaps of branches in the image and tries to reestablish the correct connectivity of the branches. However, in this process incorrect decisions are likely to be taken. For Figure 6.7a this is shown as a detail. The overlap of two branches is split into four branches here, none of them overlap, but two of them share an endpoint.



(a) RSA result

(b) geodesic distance Reeb graph incl. pruning

Figure 6.7: Results for root12, day 20. While some minor branches due to noise are wrongly kept in the final representation of the RSA tool, three small true branches are wrongly discarded in the pruning of the Reeb graph approach. The overlap of branches is wrongly dissolved by the RSA tool.

### Branch length

As the RSA tool provides root parameters such as the length of branches as well, the results of this characteristic provided by RSA were compare to both Reeb graph representations presented in this thesis. Table 6.3 shows the length of the longest branch in the root, measured in pixel from the starting point of the root to the endpoint of the branch. "RG1+P" again indicates the Reeb graph approach based on the height function with graph pruning, "RG2+P" the Reeb graph approach based on the geodesic distance with graph pruning

The length of these branches computed based on the Reeb graph representations are very similar. Due to the good alignment of the root with the vertical direction, both representations deliver similar positioned critical points. The Reeb graph representation based on the geodesic distance

Table 6.3: Overview: length of the longest branch in each root image in pixel.

| **Longest branch length** | | | |
|---|---|---|---|
| **Image** | **RG1 + P** | **RG2 + P** | **RSA** |
| root04, day 8 | 359 | 363 | 365 |
| root04, day 12 | 571 | 564 | 559 |
| root04, day 16 | 816 | 816 | 833 |
| root04, day 20 | 1111 | 1113 | × |
| root05, day 8 | 355 | 365 | 376 |
| root05, day 12 | 580 | 580 | × |
| root05, day 16 | 837 | 836 | 891 |
| root05, day 20 | 1155 | 1155 | 1171 |
| root07, day 8 | 100 | 100 | 104 |
| root07, day 12 | 118 | 118 | 112 |
| root07, day 16 | 189 | 189 | 207 |
| root07, day 20 | 232 | 232 | × |
| root09, day 8 | 123 | 126 | 127 |
| root09, day 12 | 155 | 162 | × |
| root09, day 16 | 189 | 189 | 191 |
| root09, day 20 | 316 | 316 | 346 |
| root12, day 8 | 379 | 385 | 394 |
| root12, day 12 | 585 | 590 | 589 |
| root12, day 16 | 826 | 831 | 832 |
| root12, day 20 | 1102 | 1105 | × |
| root17, day 12 | 464 | 457 | 486 |
| root17, day 16 | 703 | 710 | × |
| root17, day 20 | 971 | 971 | × |
| root19, day 8 | 395 | 395 | 423 |
| root19, day 12 | 677 | 677 | 706 |
| root19, day 16 | 967 | 967 | 1027 |
| root19, day 20 | 1312 | 1317 | × |
| root20, day 8 | 499 | 499 | 540 |
| root20, day 12 | 774 | 774 | × |
| root20, day 16 | 1076 | 1079 | × |
| root20, day 20 | 1423 | 1442 | × |
| root24, day 12 | 606 | 607 | 628 |
| root24, day 16 | 918 | 918 | 960 |
| root24, day 20 | 1257 | 1257 | × |
| **mean deviation from mean length in %** | **-1.3** | **-1.0** | **+2.3** |
| **mean deviation from mean length in pixel** | **-7** | **-6** | **+13** |

yields equal long or longer branches as the height function based representation, as it measures the intrinsic length of a root, that not only increases for vertical elongations but also for curvature. In case the branch length based on the geodesic distance is shorter than the length computed for the height function, this is due to graph pruning and the rejection of a small spurious branch at the tip of a root.

The differences in the length of the branches between Reeb graph representation and RSA are based on the different location of the start point of the root RSA may compute. RSA detects the branches in the root and allows the length measurement for such a detected branch. As the structure provided by RSA may be different than the structure based on the Reeb graph, the length of some branches could not be measured in the RSA tool. The sign "×" in the corresponding cell in Table 6.3 indicates that a measurement was not possible for this root.

Based on the set of root images, for which all three length measurements were available, the mean deviation from the mean length was computed. This deviation is ± 2.3%. The small percentage shows that the three different representations provide very similar length measurements.

## 6.4 Equality of Graphs

In order to combine the advantages of several representations, a measurement of equality of graph representations is needed. Based on this graph equality, the characteristics of various representations can be collected for a combined description of a root.

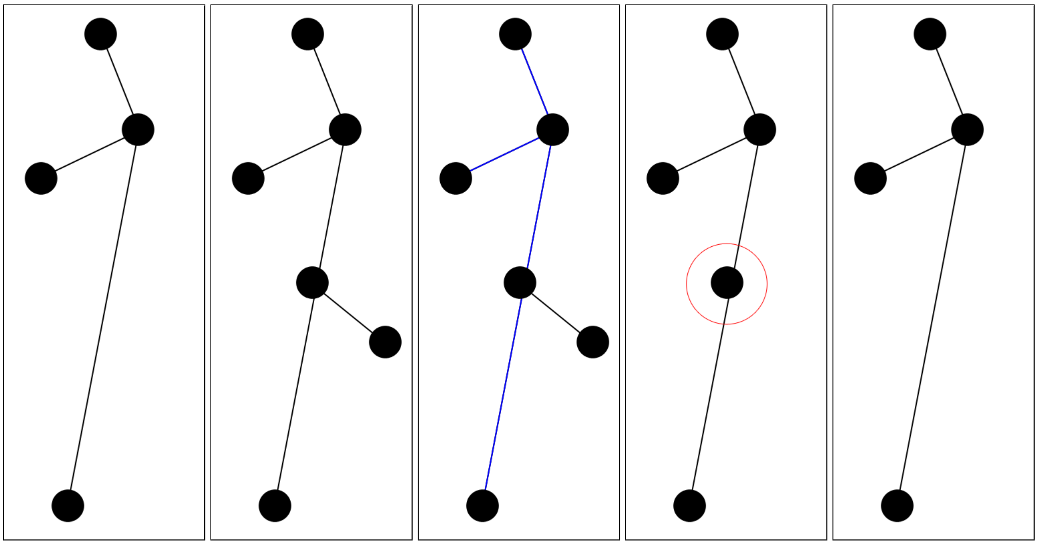
Within the scope of this thesis two graph representations of the same image are considered topologically equal, if the graphs are isomorphic. Two representations that are not equal (their graphs are not isomorphic) can be compared using a distance between these graphs. This distance can be measured based on the maximal common subgraph, as presented in [8]. For two graphs $G_1$ and $G_2$ and their maximal common subgraph $G_{mcs}$ the number of nodes in these graphs is denoted as $|G_1|$ respectively $|G_2|$ and $|G_{mcs}|$. The distance between the graphs $G_1$ and $G_2$ is defined as:

$$d(G_1, G_2) = 1 - \frac{|G_{mcs}|}{max(|G_1|, |G_2|)}. \tag{6.1}$$

The definition of subgraph and subgraph isomorphism needs to be further specified respectively altered for the scope of this thesis:

In general a subgraph is a graph built from the subset of the nodes and corresponding edges of a graph. Such a subgraph can be derived by defining a subset of the nodes and based on these nodes the corresponding subset of edges connecting the nodes. This is called a node induced subgraph. An edge induced subgraph is formed by defining a subset of edges in a graph and the endpoints form the corresponding subset of nodes. Figure 6.8 shows the process of deriving a modified subgraph. A modified subgraph is defined within the scope of this thesis as:

**Definition 20** *A modified subgraph is the edge induced subgraph which is altered by the smoothing of nodes.*



(a) graph $G_1$ (b) graph $G_2$ (c) $G_1'$ (blue) as a edge induced subgraph of $G_2$ (d) smoothing the marked node in $G_1'$ (e) $G_1$ as subgraph of $G_2$

Figure 6.8: Based on the definition in this thesis graph $G_1$ is a subgraph of $G_2$.

Based on this definition of the modified subgraph and the distance measurement given in Equation 6.1, the distance between the graphs $G_1$ and $G_2$ in Figure 6.8 is given as:
$d(G_1, G_2) = 1 - \frac{|G_1|}{|G_2|} = 1 - \frac{4}{6} = 0.33$ .

Graph isomorphism respectively subgraph isomorphism (using the modified subgraph according to Definition 20 [2]) were used to compare and evaluate the graph representations of the root dataset. Table 6.4 shows this comparison. In each case two representations of a root image are compared. Isomorphic graphs are indicated by the sign ✓in Table 6.4. All pairs of graphs that are not isomorphic, share a common subgraph. If one of the graphs already forms such a subgraph the acronym of this graph is given in the table cell. If both graphs contain isomorphic subgraphs, the corresponding cell of the table is labelled “both”.

The abbreviations used in Table 6.4 and Table 6.5 are:

- RG1 + P: Reeb graph based on height function including pruning
- RG2 + P: Reeb graph based on geodesic distance including pruning
- MA + P: Medial axis graph including pruning
- MA + WP: Medial axis graph including width based pruning

[2] Definition 20 on page 54

Table 6.4: Graph isomorphism and isomorphic subgraphs in the root dataset.

| Equality of graphs | | | | | | |
|---|---|---|---|---|---|---|
| **Image** | **RG1+P / RG2+P** | **RG1+P / MA+P** | **RG1+P / MA+WP** | **RG2+P / MA+P** | **RG2+P / MA+WP** | **MA+P / MA+WP** |
| root04, day 8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root04, day 12 | ✓ | MA+P | both | MA+P | both | MA+P |
| root04, day 16 | ✓ | MA+P | both | MA+P | both | MA+P |
| root04, day 20 | ✓ | MA+P | MA+WP | MA+P | MA+WP | ✓ |
| root05, day 8 | R2+P | ✓ | R1+P | R2+P | ✓ | MA+P |
| root05, day 12 | ✓ | R1+P | R1+P | R2+P | R2+P | MA+P |
| root05, day 16 | R1+P | ✓ | ✓ | MA+P | MA+WP | ✓ |
| root05, day 20 | ✓ | MA+P | R1+P | MA+P | R2+P | MA+P |
| root07, day 8 | ✓ | R1+P | R1+P | R2+P | R2+P | ✓ |
| root07, day 12 | ✓ | R1+P | R1+P | R2+P | R2+P | ✓ |
| root07, day 16 | ✓ | both | both | both | both | ✓ |
| root07, day 20 | ✓ | R1+P | R1+P | R2+P | R2+P | ✓ |
| root09, day 8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root09, day 12 | both | both | both | R1+P | R2+P | ✓ |
| root09, day 16 | ✓ | MA+P | ✓ | MA+P | ✓ | MA+P |
| root09, day 20 | ✓ | MA+P | R1+P | MA+P | R2+P | MA+P |
| root12, day 8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root12, day 12 | R1+P | both | both | MA+P | MA+WP | ✓ |
| root12, day 16 | ✓ | MA+P | MA+WP | MA+P | MA+WP | ✓ |
| root12, day 20 | R1+P | both | both | both | both | MA+P |
| root17, day 12 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root17, day 16 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root17, day 20 | ✓ | MA+P | both | MA+P | both | MA+P |
| root19, day 8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root19, day 12 | ✓ | MA+P | ✓ | MA+P | ✓ | MA+P |
| root19, day 16 | ✓ | MA+P | both | MA+P | both | MA+P |
| root19, day 20 | both | MA+P | ✓ | MA+P | both | MA+P |
| root20, day 8 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root20, day 12 | ✓ | MA+P | MA+WP | MA+P | MA+WP | ✓ |
| root20, day 16 | ✓ | MA+P | R1+P | MA+P | R2+P | MA+P |
| root20, day 20 | R2+P | R1+P | both | R2+P | R2+P | MA+P |
| root24, day 12 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| root24, day 16 | ✓ | ✓ | R1+P | ✓ | R2+P | MA+P |
| root24, day 20 | ✓ | MA+P | MA+WP | MA+P | MA+WP | ✓ |

Table 6.5: Topological distances between the graph representations.

| **Topological distances between the graphs** | | | | | | |
|---|---|---|---|---|---|---|
| **Image** | **RG1+P / RG2+P** | **RG1+P / MA+P** | **RG1+P / MA+WP** | **RG2+P / MA+P** | **RG2+P / MA+WP** | **MA+P / MA+WP** |
| root04, day 8 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root04, day 12 | 0.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.50 |
| root04, day 16 | 0.00 | 0.50 | 0.63 | 0.50 | 0.63 | 0.63 |
| root04, day 20 | 0.00 | 0.25 | 0.25 | 0.25 | 0.25 | 0.00 |
| root05, day 8 | 0.50 | 0.00 | 0.50 | 0.50 | 0.00 | 0.50 |
| root05, day 12 | 0.00 | 0.33 | 0.50 | 0.33 | 0.50 | 0.25 |
| root05, day 16 | 0.33 | 0.00 | 0.00 | 0.33 | 0.33 | 0.00 |
| root05, day 20 | 0.00 | 0.25 | 0.20 | 0.25 | 0.20 | 0.40 |
| root07, day 8 | 0.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.00 |
| root07, day 12 | 0.00 | 0.25 | 0.25 | 0.25 | 0.25 | 0.00 |
| root07, day 16 | 0.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.00 |
| root07, day 20 | 0.00 | 0.43 | 0.43 | 0.43 | 0.43 | 0.00 |
| root09, day 8 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root09, day 12 | 0.50 | 0.70 | 0.70 | 0.40 | 0.40 | 0.00 |
| root09, day 16 | 0.00 | 0.33 | 0.00 | 0.33 | 0.00 | 0.33 |
| root09, day 20 | 0.00 | 0.33 | 0.50 | 0.33 | 0.50 | 0.50 |
| root12, day 8 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root12, day 12 | 0.33 | 0.50 | 0.50 | 0.33 | 0.33 | 0.00 |
| root12, day 16 | 0.00 | 0.33 | 0.33 | 0.33 | 0.33 | 0.00 |
| root12, day 20 | 0.14 | 0.33 | 0.43 | 0.50 | 0.50 | 0.14 |
| root17, day 12 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root17, day 16 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root17, day 20 | 0.00 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 |
| root19, day 8 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root19, day 12 | 0.00 | 0.50 | 0.00 | 0.50 | 0.00 | 0.50 |
| root19, day 16 | 0.00 | 0.25 | 0.40 | 0.25 | 0.40 | 0.40 |
| root19, day 20 | 0.14 | 0.14 | 0.00 | 0.14 | 0.14 | 0.14 |
| root20, day 8 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root20, day 12 | 0.00 | 0.50 | 0.50 | 0.50 | 0.50 | 0.00 |
| root20, day 16 | 0.00 | 0.50 | 0.33 | 0.50 | 0.33 | 0.66 |
| root20, day 20 | 0.17 | 0.33 | 0.17 | 0.17 | 0.17 | 0.33 |
| root24, day 12 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| root24, day 16 | 0.00 | 0.00 | 0.33 | 0.00 | 0.33 | 0.33 |
| root24, day 20 | 0.00 | 0.20 | 0.20 | 0.20 | 0.20 | 0.00 |
| **Average** | **0.06** | **0.25** | **0.26** | **0.25** | **0.25** | **0.17** |

For all pairs of graphs in Table 6.4 the distance between the graphs was computed as well. These distances are given in Table 6.5. The distances of all subgraph isomorphic graphs are computed based on the distance definition given before. A distance of 0 indicates graph isomorphism.
The two Reeb graph representations based on height function and geodesic distance provide isomorphic graphs for about 80% of the root images in the dataset. For a combination with characteristics of the medial axis graphs both Reeb graph representations have an average distance to the medial axis graphs of 0.25. In relation to graph equality within the scope of this approach both Reeb graph representations yield comparable results. However, as the Reeb graphs based on the geodesic distance approximate the human ground truth graphs best, the geodesic distance will be used as the basis for the combined and the normalised representations.

## 6.5 Combined Root Representation

The three graph representations provide different characteristics of the root. The length of a root in the Reeb graph based on the height function is determined by the height of the endpoint of a root in the image. Additional length of the root due to curvature is not taken into account. This is however done for the geodesic distance, as the length is measured as the intrinsic length of the root structure. Compared to the Reeb graphs used in this thesis, the medial axis representation is the only representation, that captures the width of the roots.

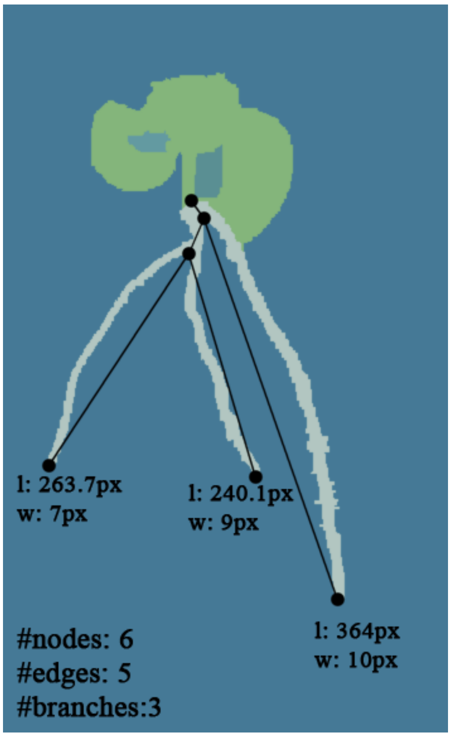


Figure 6.9: Combined Characteristics for root09, day 20.

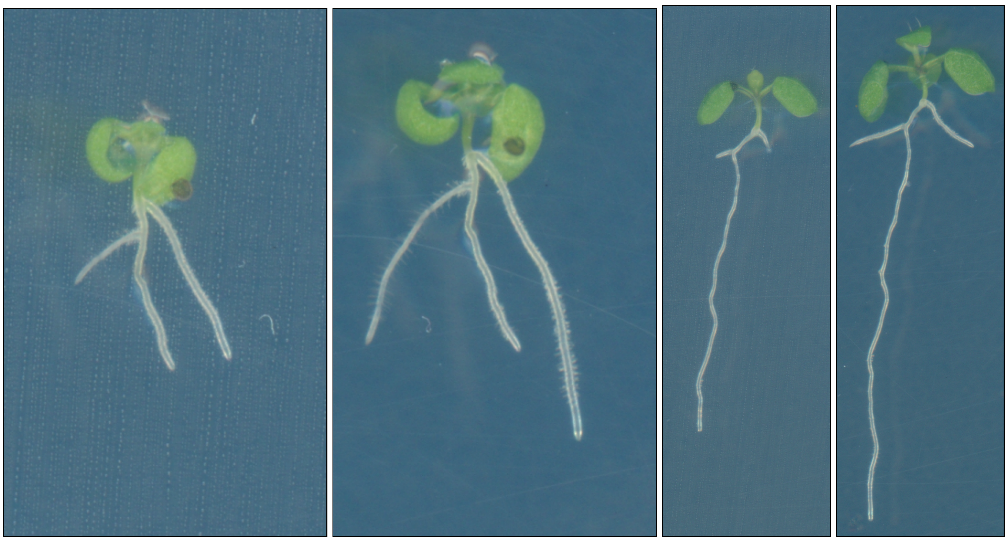
As the root images are taken as time series, several images of a root on different days of the growth cycle can be used for a normalised representation. Therefore, the images will be aligned based on the starting point of the latest image in a time series and orientated according to the longest edge from the starting point of the latest image.

In order to combine the characteristics of the three graph based representations, corresponding nodes in the graphs have to be obtained by matching of the graphs as discussed in Section 6.4. The graph is built based on one representation, here the geodesic distance based Reeb graph is a good choice, as this representation approximates the ground truth best (see Chapter 6.1). As additional information the number of nodes, branches, and the width and length of branches can be provided. Figure 6.9 shows an example for such a representation.

## 6.6 Normalised Root Representation

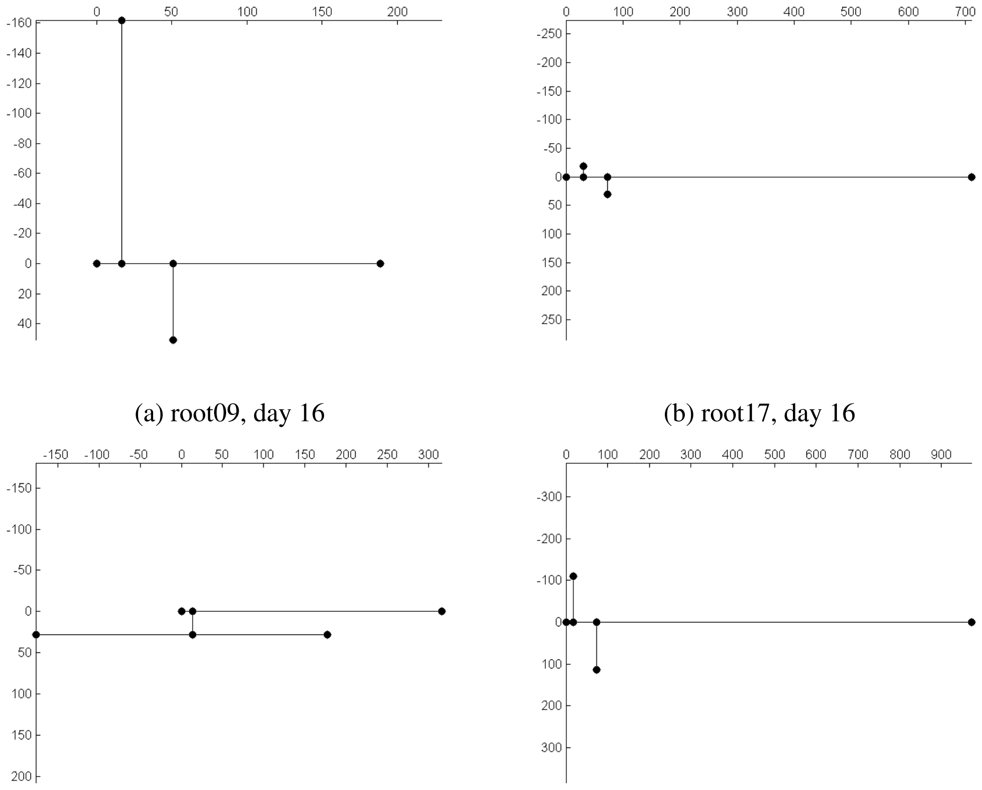
For the normalised root representation the roots are organised as branches of a main root. The main root is determined as the longest branch in the root structure. The main branch and all side branches are drawn as straight lines of their actual length. Branches occur perpendicular to the left or the right a branch. This normalised representation therefore captures the number of branches, their branching pattern, including the distances at which a branch appears, as well as the lengths of the branches. An example is shown for the four roots in Figure 6.10. The corresponding normalised representations are shown in Figure 6.11.

For root 09 the longest branch, that is taken as main root in the representation, is the branch in the middle for day 16, for day 20 it is the right branch. Therefore the normalised representations differ strongly. For root 17 both representations of day 16 and day 20 are very similar. Here



(a) root09, day 16 (b) root09, day 20 (c) root17, day 16 (d) root17, day 20

Figure 6.10: Input images for the normalised root representation.



(a) root09, day 16

(b) root17, day 16

(c) root09, day 20

(d) root17, day 20

Figure 6.11: Examples for the normalised root representation. The axis indicate the length of the branches in pixel.

the growth of the side branches is immediately visible. The difference in the distance of the branching points on the main root is due to the discretisation during the segmentation.

Figure 6.12 shows a possibility to efficiently compare the root structure of a plant on different days of the growth cycle, the individual representations are drawn as an overlay. This representation shows development of the root through the growth period very well.

For overlaps of branches in the images, this overlap is resolved by doubling the merge node for the two overlapping branches. The branches are then connected for the resolved overlap according to the angle: the branches are supposed to continue in their primary direction of growth, therefore an angle of more than 90 degrees is sought for possible connections.

This normalised representation is well suited for small branched structures as the roots in this dataset. For larger structures, with a higher number of branches, the representation may for repeated branching cause overlaps of the side branches due to the orthogonal branching representation. However, keeping the true branching angles, instead of the artificial orthogonal branching angles used for this representation, may not eliminate this problem completely: branches, that overlap in the images, still overlap in the representation. A trade-off between clear representa-

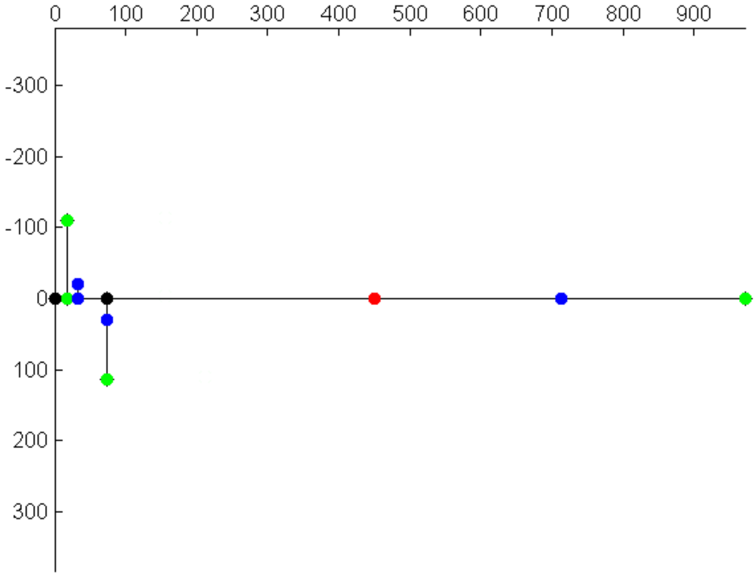


Figure 6.12: Overlay of the normalised representations for root 17. The black nodes apply to more than one graph, red indicates nodes for day 12, blue for day 16 and green for day 20. The axis indicate the length of the branches in pixel.

tion of the branching structure, correct presentation of distances and length and prevention of overlaps needs to be found depending on the represented dataset.

## 6.7 Summary of the Results

The results presented in this chapter show that the Reeb graph representations capture the root characteristics such as branching points, end points of branches and length of branches well. Comparison with human generated ground truth show that this ground truth is well approximated by the Reeb graph representations.

The different graph representations presented in this thesis are either isomorphic or share a common subgraph, this fact allows for a combination of the attributes of the subgraphs provided by the different representations.

For comparison of root images of different plants or of different days in the growth cycle a normalised representation is presented. This representation focuses on the structure of the root and therefore its branching pattern, as well as on the length of branches and the distances between branching points. It allows for an efficient comparison of the growth and development of branches of a plant on different days of the growth cycle as well for a comparison of branching patterns of different plants.

CHAPTER 7

# Conclusion

In this thesis the possibility of a Reeb graph based representation of roots was inquired. Intermediate results on the representation of root structures by Reeb graphs have been published in the Computer Vision Winter Workshop 2014 [25].

For phenotyping of plants the length of branches, numbers of branches in a root structure and the width of branches are analysed characteristics. All these traits are captured by the presented graph representations.

Two Reeb graph based approaches as well as an approach based on the medial axis were evaluated and compared to human generated ground truth. For all three approaches the graphs are based on a set of nodes and the adjacency between the nodes. The nodes are obtained by either analysing the shape according to a (Morse) function to obtain the critical points that form the nodes in the Reeb graphs or by computing a medial axis skeleton and detecting branching points and endpoints, that form the nodes of the medial axis graph. For the Reeb graph based approaches two Morse functions were used: the height function and the geodesic distance. According to the nodes, the connections in the graphs are obtained by analysing the structure. The root is traced from one critical point to the next in order to compute the adjacency of the nodes.

Reeb graphs according to a height function are able to describe a root correctly, if its general direction does not strongly deviate from the vertical direction. The length of branches in such a Reeb graph is measured considering the height function. Additional length in a branch due to curvature of the branch is not taken into account. Therefore, Reeb graphs according to the geodesic distance proved to be better suited for the task of length measurement, as in this case the intrinsic length of the root structure is measured. Reeb graphs according to the geodesic distance captured the branching pattern of the root as well as the length of individual branches.

The width of a branch is implicitly coded in the medial axis transformation. Although a graph can be built based on the medial axis as well, a combination of geodesic distance Reeb graph and medial axis should be used to correctly describe the root, as the medial axis graph (especially in combination with graph pruning) does not capture all branching points correctly.

All three representations are very sensitive to noise in the segmented image and introduce spurious branches in the graphs. 70% of these spurious branches are discarded in a graph pruning

process. However when choosing a parameter for graph pruning that removes the spurious branches, some true branches may be discarded as well. The trade-off between false positives and false negatives needs to be optimised.
As the roots are imaged through the projection of the 3D structure (the root) to the 2D image space, branches may overlap in the image. These overlaps introduce a cycle in all graph representations. One major advantage when using a Reeb graph based representation compared to a medial axis representation is, that due to the different types of nodes in a Reeb graph, an overlap generates a merge node in the graph. Therefore, an overlap can be immediately detected based on the type of the corresponding node in the graph.

The results show that the Reeb graph based representations approximate the human generated ground truth best, regarding the number of nodes in the representation as well as the structure of the graph generated.
To include parameters from several representations, as for example the width of branches based on the medial axis, the Reeb graphs can be combined with the medial axis representation. In order to compare different graph based representations, a distance based equality measurement was introduced. For isomorphic graphs the characteristics of corresponding nodes can be combined immediately, for isomorphic subgraphs this is only possible for the nodes of the subgraphs.
The segmentation, that is done as a pre-processing step, proved to be a major drawback in the attempt to automatically analyse large datasets. The semi-automatic segmentation method used for the creation of the root dataset presented in this thesis, needs to be changed to either a fully automatic approach or a semi-automatic one, that is optimised with respect to computation time and needed user interaction. This would reduce the time and human input required for the pre-processing step. Moreover, the current segmentation approach introduced artefacts in the form of frayed borders, that generate spurious branches in the graphs. Although these spurious branches can be discarded using graph pruning, true branches may be removed in the graph pruning process as well. A reduction of this noise by either using a better suited segmentation method or by application of smoothing on the segmentation results may reduce this problem.
For future work the temporal correlation of the root images should be taken into account. The roots are imaged on several days of their growth cycle, therefore the development of a root in a certain time frame is captured by such an image sequence. To distinguish a small true branch from a spurious branch may for example not be possible based on one image. A later image of the same root may help with this decision: in case there is a branch at this position in a later image, the branch is detected as a true branch, otherwise it is discarded as a spurious branch. Thus, the information and knowledge about a root gained on the images of later days in the growth cycle can be used for decisions on earlier images.

# List of Figures

# List of Tables

# Bibliography

[1] P. Armengaud, K. Zambaux, A. Hills, R. Sulpice, R. J. Pattison, M. R. Blatt, and A. Amtmann. Ez-rhizo: integrated software for the fast and accurate measurement of root system architecture. *The Plant Journal*, 57(5):945–956, 2009.

[2] J.-L. Arsenault, C. Messier S. Pouleur, and R. Guay. Winrhizo$^{\text{TM}}$, a root-measuring system with a unique overlap correction method. *HortScience*, 30(906), 1995.

[3] D. Attali, G. Sanniti di Baja, and E. Thiel. Pruning discrete and semicontinuous skeletons. In Carlo Braccini, Leila DeFloriani, and Gianni Vernazza, editors, *Image Analysis and Processing*, volume 974 of *Lecture Notes in Computer Science*, pages 488–493. Springer Berlin Heidelberg, 1995.

[4] S. Baloch, H. Krim, I. Kogan, and D. Zenkov. Rotation invariant topology coding of 2d and 3d objects using Morse theory. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–796–9, Sept 2005.

[5] S. Berretti, A. Del Bimbo, and P. Pala. 3D mesh decomposition using Reeb graphs. *Image and Vision Computing*, 27(10):1540–1554, September 2009.

[6] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Reeb graphs for shape analysis and applications. *Theoretical Computer Science*, 392(1–3):5–22, February 2008.

[7] R. Bott. Lectures on Morse theory, old and new. *Bulletin of the American Mathematical Society*, 7(2):331–358, 1982.

[8] H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19(3–4):255–259, March 1998.

[9] R. T. Clark, A. N. Famoso, K. Zhao, J. E. Shaff, E. J. Craft, C. D. Bustamante, S. R. Mccouch, D. J. Aneshansley, and L. V. Kochian. High-throughput two-dimensional root system phenotyping platform facilitates genetic analysis of root growth and development. *Plant, Cell & Environment*, 36(2):454–466, 2013.

[10] R. T. Clark, R. B. MacCurdy, J. K. Jung, J. E. Shaff, S. R. McCouch, D. J. Aneshansley, and L. V. Kochian. Three-dimensional root phenotyping with a novel imaging and software platform. *Plant Physiology*, 156(2):455–465, 2011.

[11] H. Doraiswamy and V. Natarajan. Efficient algorithms for computing Reeb graphs. *Computational Geometry*, 42(6–7):606–616, August 2009.

[12] R. EL Khoury, J. P. Vandeborre, and M. Daoudi. 3D mesh Reeb graph computation using commute-time and diffusion distances. In *Proceedings SPIE: Three-Dimensional Image Processing (3DIP) and Applications II*, volume 8290, pages 82900H–82900H–10, 2012.

[13] A. French, S. Ubeda-Tomás, T. J. Holman, M. J. Bennett, and T. Pridmore. High-throughput quantification of root growth using a novel image-analysis tool. *Plant Physiology*, 150(4):1784–1795, 2009.

[14] T. Galkovskyi, Y. Mileyko, A. Bucksch, B. Moore, O. Symonova, C. Price, C. Topp, A. Iyer-Pascuzzi, P. Zurek, S. Fang, J. Harer, P. Benfey, and J. Weitz. GiA roots: software for the high throughput analysis of plant root system architecture. *BMC Plant Biology*, 12(1):116, 2012.

[15] X. Ge, I. I. Safa, M. Belkin, and Y. Wang. Data skeletonization via Reeb graphs. In J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 837–845, 2011.

[16] M. Gerstmayer. Interactive Hierarchical Image Segmentation on Irregular Pyramids. Technical Report 129, Pattern Recognition and Image Processing Group, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2013.

[17] M. Gerstmayer, Y. Haxhimusa, and W. Kropatsch. Hierarchical interactive image segmentation using irregular pyramids. In Xiaoyi Jiang, Miquel Ferrer, and Andrea Torsello, editors, *Graph-Based Representations in Pattern Recognition*, volume 6658 of *Lecture Notes in Computer Science*, pages 245–254. Springer Berlin / Heidelberg, 2011.

[18] J. L. Gross and J. Yellen. *Graph Theory and Its Applications*. Crc Pr Inc, 2nd edition edition, August 2005.

[19] Blum H. A Transformation for Extracting New Descriptors of Shape. In Weiant Wathen-Dunn, editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, 1967.

[20] W. Harvey, Y. Wang, and R. Wenger. A randomized o (m log m) time algorithm for computing Reeb graphs of arbitrary simplicial complexes. In *Proceedings of the twenty-sixth annual symposium on Computational geometry*, pages 267–276. ACM, 2010.

[21] M. Hayashi and M. Nishimura. Arabidopsis thaliana - a model organism to study plant peroxisomes. *Biochimica et Biophysica Acta (BBA) - Molecular Cell Research*, 1763(12):1382–1391, December 2006.

[22] H. C. Hege, K. Polthier, and G. Scheuermann. *Topology-Based Methods in Visualization II*. Springer, February 2009.

[23] D. Houle, D. R. Govindaraju, and S. Omholt. Phenomics: the next challenge. *Nat Rev Genet*, 11(12):855–866, December 2010.

[24] I. Janusch. Examination of root development by means of topological image analysis. Technical Report 131, PRIP, Pattern Recognition and Image Processing Group, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2013.

[25] I. Janusch, W. G. Kropatsch, and W. Busch. Reeb graph based examination of root development. In *Proceedings of the 19th Computer Vision Winter Workshop*, pages 43–50, Feb 2014.

[26] W. Johannsen. The genotype conception of heredity. *The American Naturalist*, 45(531):pp. 129–159, 1911.

[27] W. Kropatsch, Y. Haxhimusa, and A. Ion. Multiresolution image segmentations in graph pyramids. In Abraham Kandel, Horst Bunke, and Mark Last, editors, *Applied Graph Theory in Computer Vision and Pattern Recognition*, volume 52 of *Studies in Computational Intelligence*, pages 3–41. Springer Berlin / Heidelberg, 2007.

[28] L. Lam, S. W. Lee, and C. Y Suen. Thinning methodologies-a comprehensive survey. *IEEE Transactions on pattern analysis and machine intelligence*, 14(9):869–885, 1992.

[29] Ja. Le Bot, V. Serra, J. Fabre, X. Draye, S. Adamowicz, and L. Pagès. Dart: a software to analyse root system architecture and development from captured images. *Plant and Soil*, 326(1-2):261–273, 2010.

[30] D. T. Lee. Medial axis transformation of a planar shape. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-4(4):363–369, July 1982.

[31] D. Leitner, B. Felderer, P. Vontobel, and A. Schnepf. Recovering root system traits using image analysis exemplified by two-dimensional neutron radiography images of lupine. *Plant Physiology*, 164(1):24–35, 2014.

[32] G. Lobet, L. Pagès, and X. Draye. A novel image-analysis toolbox enabling quantitative analysis of root system architecture. *Plant Physiology*, 157(1):29–39, 2011.

[33] S. Mairhofer, S. Zappala, S. R. Tracy, C. Sturrock, M. Bennett, S. J. Mooney, and T. Pridmore. Rootrak: Automated recovery of three-dimensional plant root architecture in soil from x-ray microcomputed tomography images using visual tracking. *Plant Physiology*, 158(2):561–569, 2012.

[34] I. Miko. Genetic dominance: genotype-phenotype relationships. *Nature Education*, 1(1):140, 2008.

[35] K. A. Nagel, A. Putz, F. Gilmer, K. Heinz, A. Fischbach, J. Pfeifer, M. Faget, S. Blossfeld, M. Ernst, C. Dimaki, B. Kastenholz, A. K. Kleinert, A. Galinski, H. Scharr, F. Fiorani, and U. Schurr. GROWSCREEN-Rhizo is a novel phenotyping robot enabling simultaneous measurements of root and shoot growth for plants grown in soil-filled rhizotrons. *Functional Plant Biology*, 2012.

[36] M. Natali, S. Biasotti, G. Patanè, and B. Falcidieno. Graph-based representations of point clouds. *Graph. Models*, 73(5):151–164, September 2011.

[37] A. Pierret, S. Gonkhamdee, C. Jourdan, and J. L. Maeght. Ij_rhizo: an open-source software to measure scanned images of root samples. *Plant and Soil*, 373(1-2):531–539, 2013.

[38] M. P. Pound, A. P. French, J. A. Atkinson, D. M. Wells, M. J. Bennett, and T. Pridmore. Rootnav: Navigating images of complex root architectures. *Plant Physiology*, 162(4):1802–1814, 2013.

[39] D. Ristova, U. Rosas, G. Krouk, S. Ruffel, K. D. Birnbaum, and G. M. Coruzzi. Rootscape: A landmark-based system for rapid screening of root architecture in arabidopsis. *Plant Physiology*, 161(3):1086–1096, 2013.

[40] R. Sozzani and Benfey P. N. High-throughput phenotyping of multicellular organisms: finding the link between genotype and phenotype. *Genome Biology*, 12(3):219, 2011.

[41] J. Stewart. *Calculus*. Cengage Learning Emea, 6th edition. international met edition, February 2008.

[42] C. M. van der Weele, W. G. Spollen, R. E. Sharp, and T. I. Baskin. Growth of Arabidopsis thaliana seedlings under water deficit studied by control of water potential in nutrient-agar media. *Journal of Experimental Botany*, 51(350):1555–1562, 2000.

[43] A. Vital Saúde, M. Couprie, and R. Lotufo. Exact euclidean medial axis in higher resolution. In Attila Kuba, LászlóG. Nyúl, and Kálmán Palágyi, editors, *Discrete Geometry for Computer Imagery*, volume 4245 of *Lecture Notes in Computer Science*, pages 605–616. Springer Berlin Heidelberg, 2006.

[44] N. Werghi, Y. Xiao, and J. P. Siebert. A functional-based segmentation of human body scans in arbitrary postures. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(1):153–165, 2006.

[45] G. Zeng, S. T. Birchfield, and C. E. Wells. Automatic discrimination of fine roots in minirhizotron images. *New Phytologist*, 177(2):549–557, 2008.

[46] Y. Zheng, S. Gu, H. Edelsbrunner, C. Tomasi, and P. Benfey. Detailed reconstruction of 3D plant root shape. In *2011 IEEE International Conference on Computer Vision (ICCV)*, pages 2026–2033, 2011.