MSc thesis in Geomatics

Plant Skeleton Extraction and Stem-leaf Segmentation

Qiwei Shen

2024

MSc thesis in Geomatics

Plant Skeleton Extraction and Stem-leaf Segmentation

Qiwei Shen

October 2024

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

Qiwei Shen: *Plant Skeleton Extraction and Stem-leaf Segmentation* (2024)

The work in this thesis was carried out in the:

TU Delft
3Dgeoinfo

3D geoinformation group
Delft University of Technology

Supervisors:	Prof.dr. Liangliang Nan
	Prof.dr. Jantien Stoter
Co-reader:	Prof.dr. Roderik Lindenbergh

Abstract

Plant phenotyping plays a vital role in plant genetics and breeding programs, providing the foundation for screening and evaluating genetic diversity and linking phenotypic parameters to the genetic determinants of trait expression. This process is critical for identifying molecular markers and accelerating genetic breeding improvement, thereby enhancing plant resilience to biotic and abiotic stresses such as drought, salinity, and diseases. Recent advancements in 3D sensing technology have empowered researchers to extract precise phenotypic parameters from plant point clouds, enabling more detailed and accurate plant phenotyping. A critical step in point cloud-based plant phenotyping is plant organ segmentation. Among the available segmentation methods, skeleton-based approaches are simple and intuitive, and could leveraging both local and global geometric information from plant point clouds to facilitate accurate organ segmentation. These methods have gained considerable attention in recent years. However, plant skeletonization, the core component of these approaches, remains limited in handling leafy plants, especially herbaceous species with complex shoot architectures, lateral stems, and multiple leaves. These structural complexities pose challenges that current skeletonization techniques struggle to address effectively.

To address this challenge, we propose a skeleton-based plant organ segmentation framework that accurately extracts curve skeletons from individual plant point clouds and performs precise stem-leaf segmentation based on these skeletons. Our framework is particularly effective in handling leafy plants. It preserves fine structural details during skeletonization while avoiding abnormal or noisy local branches by extending the Laplacian-Based Contraction (LBC) algorithm through the integration of the Constrained Laplacian Operator. Moreover, we introduce Adaptive Constraints and Tip Points Preservation within the contraction loops to further refine skeleton quality. Additionally, a modified Locally Optimal Projection (LOP) operator is utilized to perform skeleton points calibration, ensuring that the extracted skeleton is centrally aligned with the original plant shape. Furthermore, to evaluate the performance of our proposed framework, we contribute a photogrammetric 3D plant point cloud dataset of 56 ***Polygonum lapathifolium*** plants, complete with detailed annotations. Experiment results demonstrate that our framework robustly handles various shapes and sizes of leafy plants and tree branches.

In conclusion, our study enhances the LBC algorithm by integrating the Constrained Laplacian Operator, Adaptive Constraints, and Tip Points Preservation. These improvements increase the accuracy and quality of curve skeleton extraction from leafy plant point clouds, enabling satisfactory plant organ segmentation.

Acknowledgements

With the completion of this thesis, my journey as a Master student in TU Delft has come to an end. These past 26 months have felt like an exhilarating journey for me—challenging, at times overwhelming, but ultimately incredibly rewarding. I still vividly remember stepping off the plane at Schiphol Airport on August 24th, 2022. Filled with enthusiasm, I was eager to transition from my bachelor's background in plant and soil sciences to a new and challenging field, Geomatics. As I reflect on this journey, my heart is filled with profound gratitude and emotion. This experience has not only expanded my academic horizons but also taught me the value of perseverance and the importance of a supportive community.

Reflecting on this graduation project, my heart overflows with profound gratitude for my first supervisor, Prof. Liangliang Nan. His unwavering trust and constant encouragement became the bedrock upon which I tackled every challenge and question that surfaced during my research. Since July 19th, 2023, we have had about 30 meetings; without his endless support and belief in me, I could not have reached this point. I especially remember the challenging period between May and June 2024, when I felt lost in a maze of complex optimization problems and nearly lost the motivation to continue this project. Yet, it was Prof. Liangliang's critical scientific attitude and relentless pursuit of truth that reignited my spirit. Like a searchlight through the fog, his guidance illuminated a path I thought had vanished. His mentorship rekindled my passion and confidence, enabling me to overcome obstacles I once deemed insurmountable. From the depths of my heart, I express my sincerest appreciation for his guidance. His support has not only been instrumental in the completion of this thesis but has also profoundly shaped my approach to scientific inquiry.

My deepest thanks also go to my second supervisor, Prof. Jantien Stoter, for her invaluable suggestions and steadfast support throughout this journey. Her insights and expertise were crucial in steering the direction of this thesis, and I cannot thank her enough for her contributions. The entire process was a formidable challenge, but with her encouragement and wisdom, I was able to navigate the complexities and evolve both personally and professionally.

I would like to thank my co-reader, Prof. Roderik Lindenbergh, and the Delegate of the Examiners, Daniëlle Groetelaers. Their careful examination and enlightening suggestions have effectively enhanced the integrity and depth of this thesis. I express my sincere thanks to them for their kind arrangements, inspired suggestions, and time.

Additionally, I would also like to thank Prof. Tingxuan Li at Sichuan Agricultural University, College of Resources. Without his kind permission and support, I could not have obtained the plant materials for 3D reconstruction, which is crucial for the whole project.

I want to express my deepest gratitude to my parents and Yanchun Li. Without our daily WeChat video calls, I would have struggled to cope with my homesickness and the challenges of studying abroad. Your unwavering support and love have been my anchor throughout this journey. Thank you for always being there for me.

Finally, I wish to extend my heartfelt thanks to all those who have supported me during this journey. Your encouragement and assistance have been indispensable, and I could not have reached this point without you. Thank you for being part of this significant chapter in my life.

Qiwei Shen

Delft

October, 2024

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

GWASs	Genome-Wide Association Studies	1
ToF	Time-of-Flight	2
SfM	Structure from Motion	2
MVS	Multi-View Stereo	2
LiDAR	Light Detection and Ranging	2
NeRFs	Neural Radiance Fields	2
LBC	Laplacian-Based Contraction	v
DBSCAN	Density-Based Spatial Clustering of Applications with Noise	8
RANSAC	Random Sample Consensus	9
MST	Minimum Spanning Tree	xi
LOP	Locally Optimal Projection	v

1 Introduction

1.1 Background and Motivation

Plant phenotyping involves the quantitative acquisition, modeling, and analysis of plant traits regulated by the dynamic genotype-by-environment interactions [Walter et al., 2015; Yang et al., 2020]. This quantitative study on plant trait diversity is crucial for Genome-Wide Association Studies (GWASs). It could provide a foundation to identify genetic architectures and accelerate genetic-based breeding improvement, thereby increasing plant yield and enhancing plant resilience to biotic and abiotic stresses, such as drought, salinity, and plant diseases [Xiao et al., 2017, 2022; Das Choudhury et al., 2019; Krishna et al., 2023]. Therefore, this is vital for ensuring global food security amidst climate change and soil degradation. Unlike the advanced high-throughput DNA sequencing technology in plant genomics, in the plant phenomics sector, the plant phenotyping approaches mainly remain in their infancy stage [Hu et al., 2021]. Confronted with a large number of plant materials, the lack of efficient high-throughput plant phenotyping technology to acquire reliable phenotypic data has emerged as a bottleneck for agronomy and plant sciences [Mir et al., 2019].

Traditional plant phenotyping approaches, marked by their labor-intensive, time-consuming, and frequently invasive nature and their dependence on field observations and manual measurements by researchers, are becoming increasingly insufficient to meet the requirements of modern plant phenomics studies. These methods represent a significant gap in associating phenotypic traits with genotypic data, thereby hindering the efficiency of developing new cultivars and advancing the understanding of plant functional genomics [Tardieu et al., 2017; Liu and Yan, 2018; Deery et al., 2016]. Since the last several decades, with the rapid development of sensing technologies, the integration of computer vision in plant phenotyping has advanced significantly, which is evident in exploring high-throughput, non-invasive, and multi-scale plant traits acquisition pipelines [Berger et al., 2010; Das Choudhury et al., 2019; Zhang et al., 2023]. Therefore, to break the "phenotypic bottleneck", the development of computer vision-based, automated, and high-throughput plant phenotyping method has attracted much attention from both agriculture and computer science communities.

Over the past few decades, much research has focused on extracting plant morphology information at the individual plant scale (e.g., plant height, width, and volume) and complex traits at the organ level (e.g., leaf area, stem length, and angles of leaves-stem) by utilizing computer-vision techniques [Paulus, 2019]. These efforts are important for accurately assessing plant growth, health, and yield, which have the potential to promote and guide the development of new plant breeding lines [Jimenez-Berni et al., 2018]. Quantitative analysis of plant architecture and organ phenotypic traits is typically based on the individual components of the plant. Thereby, identifying plant organs, as facilitated by stem-leaf segmentation from captured data, is one of the crucial tasks in phenotyping, as it forms the foundation for calculating detailed phenotypic parameters.

Consequently, much work and effects have been done to perform plant organ segmentation from 2D images. These 2D image-based organ segmentation approaches, whether automatic or semi-automatic, have increased efficiency and reduced the manual labor required in conventional plant phenotyping practices. However, the lack of depth information makes 2D image-based phenotyping approaches challenging to address issues, such as plant organ occlusions and crossover, and often fail to extract accurate areal and volumetric data, which makes it hard to describe plants with complex morphological structures accurately [Shi et al., 2019; Sun et al., 2020].

Those limitations and the recent significant strides made by 3D sensing technologies in fields such as autonomous driving and robotics have prompted increasing attention to the application and exploration of 3D plant phenotyping methods [Guo et al., 2021b; Harandi et al., 2023]. By reconstructing high-quality 3D plant point cloud model through Time-of-Flight (ToF) cameras [Xiang et al., 2019; Ma et al., 2021b; Xu et al., 2023], Structure from Motion (SfM) and Multi-View Stereo (MVS) methods [Rose et al., 2015; Li et al., 2022c; Liu et al., 2022; Yang et al., 2024], 3D laser scanners (Light Detection and Ranging (LiDAR)) [Paulus et al., 2013; Jin et al., 2021; Forero et al., 2022], or Neural Radiance Fields (NeRFs) algorithms [Hu et al., 2024; Meyer et al., 2024], scholars are provided with the necessary basis to extract and describe plant organ instances, canopy architecture, precisely. Thus, 3D plant point clouds can potentially improve the quality of the extracted phenotypic parameters [Jimenez-Berni et al., 2018; Forero et al., 2022].

The common conventional approaches to segment plant organs from 3D point clouds mainly include clustering and model fitting-based methods. Clustering-based methods generally use the difference in local geometric surface feature, density, and distance on different plant organ point clouds as the clustering criterion to segment the points into clusters. On the other hand, model fitting-based methods leverage the distinct morphological features of plant stems and leaves, characterized by their linear and planar structures, respectively, to accurately identify these structures by fitting the points to customized geometric shapes.

While both approaches can be practical, they heavily rely on selecting suitable local geometric features and often necessitate further refinement to achieve optimal segmentation results. Additionally, these methods tend to be noise-sensitive and require accurate initial parameter determination. Poor parameter selection can result in sub-optimal segmentation results, particularly when dealing with plant species with significantly inherent variability in the appearance and shape [Raykov et al., 2016]. Moreover, the complexity of plant morphology and structure makes these methods time-consuming and labor-intensive, often requiring significant manual intervention and parameter tuning, as well as a deep understanding of plant morphology for the operators [Patel et al., 2023; Li et al., 2022b]. This limitation presents challenges in managing the large volumes of data that are typically involved in high-throughput plant phenotyping [Jin et al., 2019].

Furthermore, recent studies have demonstrated that deep learning-based methods exhibit high generalizability and accuracy in various 3D point cloud tasks, such as object classification, tracking, and segmentation [Guo et al., 2021b]. Leveraging these methods for plant organ segmentation presents an opportunity to overcome the limitations of traditional 3D plant organ segmentation techniques. However, deep learning-based methods are not without drawbacks. While deep learning-based methods can obtain high segmentation accuracy and efficiency, they require a large amount of labeled data for model training [Wang et al., 2024]. The manual annotation of 3D point clouds is a highly labor-intensive and time-consuming process that demands specialized plant knowledge. Consequently, deploying these methods in complex, real-world environments is challenging and costly.

In addition to these approaches, much work has been done on skeleton-based methods, which aim to segment plant organs or extract more phenotypic parameters based on the curve skeleton of the plant simountenously [Bucksch, 2014]. The curve skeleton, a simplified and compact representation of the geometry and topology of a 3D point cloud model, can intuitively reflect both morphological and topological information [Au et al., 2008; Chaudhury and Godin, 2020]. Therefore, unlike clustering and model fitting-based approaches, which primarily focus on local geometric information, skeleton-based methods can leverage both the local and global topological structures of plant point clouds for organ segmentation [Wang et al., 2023]. Due to these characteristics, plant skeletons, which encode the positions of plant organs and architectural traits, have been employed in plant architecture analysis and organ segmentation tasks for several decades.

However, extracting curve skeletons from 3D data still remains an “open question” [Tagliasacchi et al., 2016]. To enable precise plant organ segmentation based on the skeleton, the extracted skeleton should meet several criteria, including both topological and geometrical accuracy, preservation of the overall shape (i.e., integrity), correct location, and correct connectivity among skeleton branches [Wu et al., 2021]. The work of Ma et al. [2021a] and Ma et al. [2023] successfully adapted the L_1-medial skeleton algorithm to extract plant skeletons and achieved satisfactory organ segmentation results by decomposing the extracted skeletons. While the L_1-medial skeleton algorithm is effective in handling complex cylindrical structures and maintaining the centeredness of the skeleton, it is less suitable

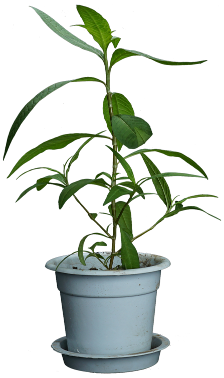


Figure 1.1: Appearance of ***P. lapathifolium*** plant.

for flat or planar structures, such as wide leaves [Wu et al., 2019], and it often requires complicate parameter tuning.

The LBC algorithm has been proven to be effective in extracting curve skeletons from both cylindrical objects [Li et al., 2020] and surfaces with boundaries, such as plant stems and leaves. As a result, it has become a widely utilized approach for plant skeleton extraction [Wu et al., 2019; Miao et al., 2021]. It is worth noting that the performance of LBC algorithm heavily depends on the quality of the constructed Laplacian operator. However, due to the non-manifold nature of point clouds, maintaining the quality of the Laplacian operator during contraction interactions is challenging and often leads to abnormal contraction directions [Meyer et al., 2003; Sharp and Crane, 2020]. These abnormal contraction directions can cause the skeleton to deviate from the center lines, resulting in zigzag structures that may even extend outside the original point cloud. Such deviations can result in incomplete or inaccurate skeletons, which, when applied to plant organ segmentation, reduce the efficiency and accuracy of the segmentation results. Additionally, preserving small branches and tiny leaves in the resulting skeleton is difficult using the LBC algorithm. Therefore, interactive corrections or the imposition of geometric constraints are frequently required to ensure the quality of the skeleton extracted by LBC algorithm.

Previous research has identified several limitations in current plant skeleton extraction algorithms. These algorithms often struggle with plant leaves, performing poorly in mitigating zigzag structures, preserving fine details, maintaining centrality and accurately representing the structural features of plants. Additionally, there is still a lack of research on algorithms that are capable of extracting satisfactory skeletons from leafy plants. To address this research gap, this study aims to develop a framework to process individual leafy plant point clouds, accurately extract plant curve skeletons, and perform stem-leaf segmentation based on the extracted skeleton. The framework is expected to perform particularly well on herbaceous plants, which typically exhibit complex shoot structures with lateral stems and multiple leaves. Therefore, in this study, we select the seedlings of ***Polygonum lapathifolium***, which are annual herbaceous plants with relative complex shoot structures, widely distributed in Sichuan, China, as our research plants (as shown in Figure 1.1).

We propose refining the LBC algorithm to improve performance by introducing geometric constraints and optimization operations. These enhancements will allow us to extract curve skeletons that accurately approximate the topological and geometric features of the original complex leafy plant point cloud while maintaining curve shape, ensuring centrality, and eliminating abnormal branches and zigzag structures. Finally, we will compare the performance of our refined plant skeleton extraction algorithm with state-of-the-art algorithms on different plant species and evaluate the performance of stem-leaf segmentation, which is based on our extracted skeleton, using ***P. lapathifolium*** point cloud data.

1.2 Research Objectives and Questions

The primary objective of this study is to develop a framework that integrates plant skeleton extraction and stem-leaf segmentation. This framework aims to automatically compute a satisfactory curve skeleton and, based on the skeleton, achieve accurate segmentation results from a plant point cloud. The curve skeleton should maintain both topological and geometrical accuracy while preserving local fine geometric features and centeredness.

Following these research objectives, our research questions can be formulated as follows:

- How can we maintain correct contraction directions while preserving both local and global geometric information?
- How can we contract a leafy plant point cloud completely into a skeletal object without leaving significant under-contraction areas?
- How can we ensure the extracted skeleton preserves the overall plant shape effectively (i.e., without significant gaps between skeleton points)?
- How can we ensure that the shape of the extracted skeleton fits the original point cloud well?
- How effective is our refined LBC algorithm for plant stem-leaf segmentation?

1.3 Research Scope and Challenges

This study focuses on developing a set of geometric constraints that can be applied to the LBC algorithm to obtain a satisfactory plant curve skeleton that meets the criteria, including mitigating zigzag structures, preserving fine details, maintaining centrality, and accurately representing the structural features of plants. Following this, we will establish plant stem-leaf segmentation based on the extracted curve skeleton. The scope of this research is limited to plant skeleton extraction and stem-leaf segmentation. Therefore, we assume that the input plant is represented as an individual point cloud, which has already been segmented from the background and properly denoised.

Previous studies have primarily focused on extracting plant curve skeletons from plants with simple structures, typically characterized by a single main stem, few or no lateral stems, and minimal or no leaves. In contrast, ***P. lapathifolium***, a herbaceous plant, exhibits a more complex and varied shoot structure (Fig. 1.1). Compared to plants like maize, sorghum, or tree branches studied in earlier research, our research herbaceous plant, ***P. lapathifolium*** presents a significantly more intricate appearance and structure, making the extraction of a satisfactory curve skeleton from its point clouds a challenging task.

However, a successful solution to this challenge could lead to high accuracy in leafy plant organ segmentation and enhance the quality of other structural phenotypic trait extractions. Therefore, developing a plant skeleton extraction algorithm based on a refinement of the LBC algorithm, capable of handling herbaceous plants with complex shoot structures (including lateral stems and multiple leaves), represents a valuable research problem. Thus, to achieve this goal, several challenges must be addressed in this study.

1.4 Thesis Outline

The thesis is organized as follows:

- Chapter 2 gives a review of related works, beginning with common 2D image-based plant organ segmentation approaches, followed by relevant studies in 3D plant point cloud segmentation. It also includes several key works on plant skeleton extraction algorithms.

- Chapter 3 details the entire pipeline of our proposed skeleton extraction and stem-leaf segmentation algorithms. It introduces our designed geometric constraints and explains their mechanisms. The strategy for stem-leaf segmentation is also introduced. Additionally, this chapter presents the pipeline for reconstructing ***P. lapathifolium*** plant point clouds, which form our own dataset. This dataset can be used to evaluate the performance of our framework on leafy plants. It also covers the evaluation criteria for stem-leaf segmentation.
- Chapter 4 explains the implementation details, including the tools and parameters used, demonstrates the results of plant skeleton extraction and stem-leaf segmentation, and provides a comprehensive evaluation and analysis of the proposed framework. Comparisons between our refined plant skeletonization algorithm and other state-of-the-art methods, and its performance on other plant species are also presented. Relevant discussions on parameter selection, limitations, and potential applications are provided.
- Chapter 5 provides a conclusion and outlines future work.

2 Related work

In this chapter, we first list the common 2D image-based plant organ segmentation techniques in Section 2.1. Then, in Section 2.2, we cover an overview of existing plant organ segmentation approaches from 3D point clouds, including clustering-based, model fitting-based, deep learning-based and skeleton-based approaches. Last, several research and studies focused on plant skeletonization are discussed in Section 2.3.

2.1 2D Image-based Plant Organ Segmentation

In the last several decades, the integration of computer vision in plant phenotyping has advanced significantly, which is evident in the efficient extraction of plant traits and reduction of manual labor [Das Choudhury et al., 2019]. Among those computer vision-based plant phenotyping pipelines, the importance of high-precision segmentation of plant organs is self-evident. Consequently, much work and effort have been done to develop 2D image-based plant organ segmentation approaches.

For instance, Pape and Klukas [2015] developed a method for leaf instance segmentation from rosette plants by combining distance map-based leaf center detection and leaf split points detection. In a similar line of work, Scharr et al. [2015] introduced a method that combined a super-pixel approach with a distance map that can efficiently segment the leaves of *Arabidopsis* and tobacco. Zhang et al. [2017] achieved maize stem-leaf identification by extracting and segmenting the 2D curve skeleton from binary images using the parallel thinning algorithm and the Hough transformation. Similarly, Das Choudhury et al. [2018] employed the fast marching distance transform method to generate a 2D plant skeleton graph from maize images and then successfully segmented individual components of maize by decomposing the skeleton graph according to the node characteristics. Furthermore, Yin et al. [2018] proposed a multi-leaf segmentation algorithm that utilized optimal leaf edge map candidates selection and Chamfer matching. In more recent work, Praveen Kumar and Domnic [2019] proposed a strategy to construct a plant graph model through enhanced imaging and a graph algorithm and achieve leaf instance identification.

Those automated and semi-automated 2D image-based organ segmentation techniques have significantly improved the efficiency of traditional phenotyping methods by reducing the need for manual labor. Nevertheless, the inherent lack of depth information in 2D approaches presents considerable challenges, particularly in addressing organ occlusions and overlaps. Moreover, these methods often struggle to perform accurate areal and volumetric measurements, making them less effective in representing plants with complex morphological structures [Shi et al., 2019; Sun et al., 2020].

2.2 Plant Organ Segmentation

2.2.1 Clustering-based approaches

Clustering-based methods for 3D plant point cloud segmentation typically rely on the local geometric surface features, point density, and distance differences between different plant organs as the key clustering criteria.

Surface feature histograms, which encode geometric information by analyzing point neighborhoods and surface normals, have been developed to recognize geometric primitives in 3D point clouds, such as planes, cylinders, and spheres. Plant organs, such as leaves and stems, tend to generate unique surface feature histograms due to their alignment with these geometric primitives, thus aiding in effective segmentation. For instance, Wahabzada et al. [2015] introduced a method that used surface feature histograms as local geometry descriptors for each point. They employed the Hellinger distance for clustering to mitigate noise-related issues and successfully separated leaves and stems across three plant species (grapevine, wheat, and barley) using the *k*-means clustering algorithm.

However, the performance of this method is sensitive to the calculation parameters used in histogram computation, and the choice of similarity measures among histograms also plays a critical role. Consequently, parameter tuning or specific refinement is often required, particularly when dealing with different plant species or one plant species but with varying shape appearances.

Beyond surface feature histograms, first- and second-order tensors have also been employed to encode local point properties. Based on this, Elnashef et al. [2019] developed a segmentation model for cotton, maize, and wheat plants. First, they obtained those plant point clouds by reconstructing them from a series of 2D images via the MVS photogrammetric 3D reconstruction technique. Then, They computed these tensors to describe the local geometric properties of each point in the 3D plant point cloud. By analyzing the differences in tensor values across various plant organs, Elnashef et al. [2019] achieved successful classification of stem and leaf points in cotton, maize, and wheat, with accuracy rates of 97%, 96%, and 93%, respectively. They further used the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm to implement leaf instance segmentation.

In more recent work by Wang et al. [2023], they utilized the Minkowski distance field to represent geometric features of the maize point cloud and segmented organ instances by executing Quickshift++ [Jiang et al., 2018] on that distance field. An average precision of 88% was achieved in leaf instance separation.

Similar to surface feature histograms, the performance of tensor-based and distance-based methods is also sensitive to parameters, such as the search radius and thresholds for distinguishing between points of different plant organs. Inappropriate parameter selection can lead to unsatisfactory classification outcomes.

2.2.2 Model fitting-based approaches

Model fitting-based methods leverage the distinct morphological features of different plant organs, characterized by their linear, planar, or sphere structures (e.g., stems, leaves, and fruits). These methods identify these structures by fitting the points into customized geometric shapes.

Using MVS reconstruction techniques and digitization operations, Paproki et al. [2012] obtained high-quality 3D plant mesh models. They proposed a pipeline to segment stems, petioles, and leaves automatically by combining a region-growing algorithm with a model-fitting approach (see as in Figure 2.1). Based on the coarse segmentation results obtained from the region-growing algorithm, they accomplished accurate stem and petiole segmentation using a model-fitting algorithm with the designed tubular shape geometry primitives.

Chaivivatrakul et al. [2014] employed ToF cameras and the MVS reconstruction method to generate point clouds of maize plants. They successfully identified maize stems by vertically slicing the maize point cloud and applying least squares ellipse fitting to detect slices that belong to the stem. After removing the detected stem, they achieved leaf instance segmentation from a 2D image projected by the point cloud by finding contours after morphological operations.

A similar approach was conducted by Gelard et al. [2017]; they also performed leaf instance segmentation based on a point cloud after removing the main stem. They improved the traditional cylinder-fitting algorithm to detect and remove the stem by introducing a “climbing ring” strategy. In this method, a ring with a user-defined radius, tailored to the test plant, starts from the base and climbs along the stem, constrained by both local neighborhood and normal constraints. By monitoring changes in the normal direction of those rings, the rings located on the main stem can be identified. The fusion of these rings forms the detection of the main stem. After removing the points

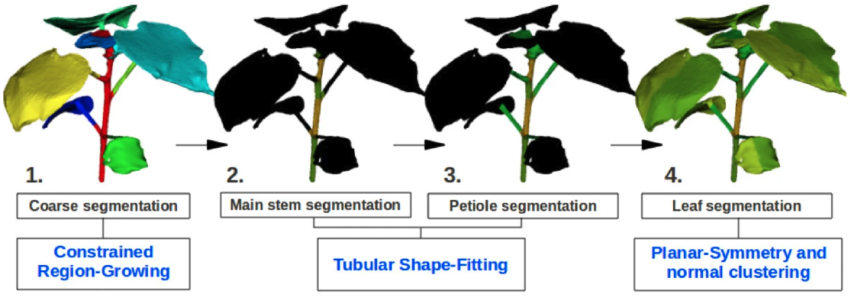


Figure 2.1: Model fitting-based plant organ segmentation pipeline [Paproki et al., 2012].

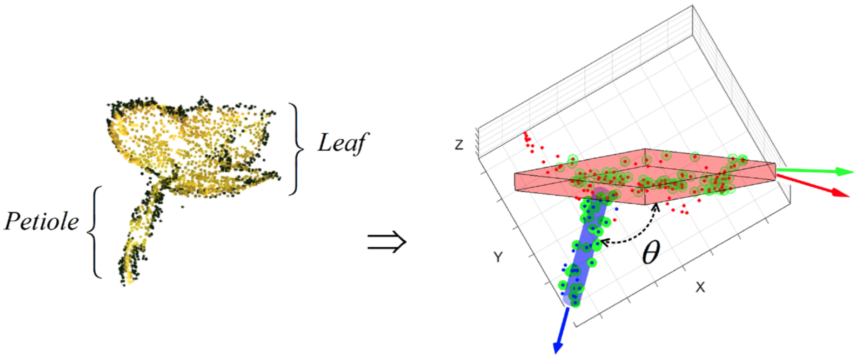


Figure 2.2: Model fitting-based approach for leaf-petiole segmentation [Ghahremani et al., 2021].

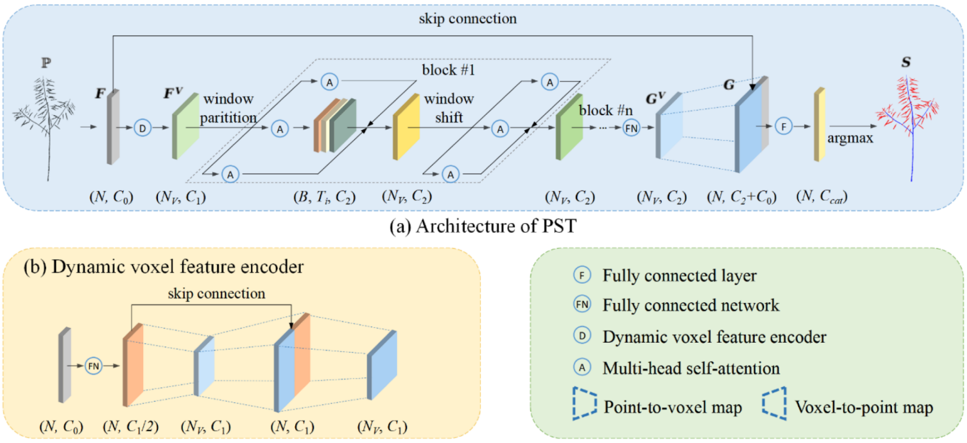
within the identified rings, the main stem is effectively removed, enabling subsequent leaf instance segmentation using a distance-based clustering algorithm.

Ghahremani et al. [2021] employed the Random Sample Consensus (RANSAC) algorithm for modeling different plant organs. They first designed different geometric shapes, such as cuboids and cylinders, according to specific plant parts. The RANSAC algorithm was then used to fit these models onto the respective parts of the input point cloud, facilitating the identification of different plant organs.

Although the aforementioned model-fitting approaches are simple and often efficient, they heavily depend on selecting appropriate parameters, such as those defining user-specified geometric primitives. This process often requires prior knowledge of plant architecture and may necessitate further refinement to achieve optimal segmentation results.

2.2.3 Deep learning-based approaches

In recent studies, deep learning-based methods have demonstrated high generalizability and accuracy in various 3D point cloud tasks, such as shape classification, object tracking, and semantic/instance segmentation [Guo et al., 2021b]. In this context, deep learning methods for plant organ segmentation offer promising solutions to the limitations of the above conventional segmentation techniques. Over the past five years, Transformers-based architecture models [Vaswani et al., 2017] have shown impressive results in 3D point cloud segmentation in ordinary scenes [Zhao et al., 2021; Guo et al., 2021a], and they have gradually been adapted in the development of deep learning-based algorithms for plant organ segmentation. The following summarizes several novel deep learning-based approaches built on Transformer architectures.



(a) Architecture of PST

Figure 2.3: The architecture of PST [Du et al., 2023].

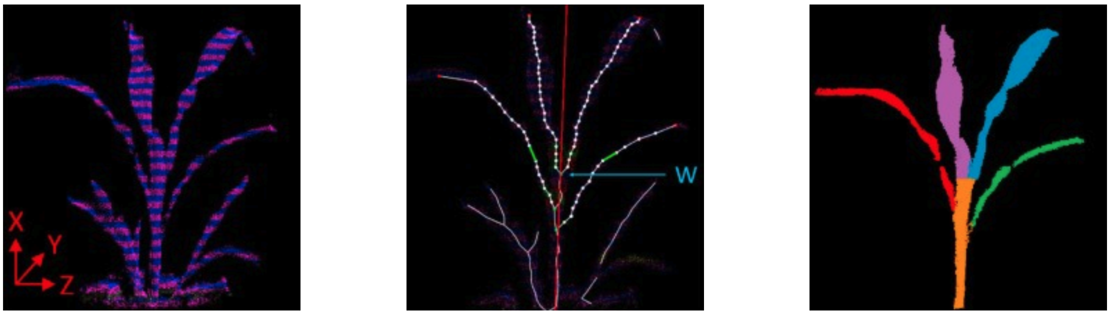
For example, Li et al. [2022b] introduced PlantNet, a point-wise network with an average precision of more than 90% in stem-leaf segmentation for tobacco, tomato, and sorghum. Li et al. [2022a] presented PSegNet, an advanced network for plant point cloud segmentation, featuring a Double-Granularity Feature Fusion Module and Attention Modules with spatial and channel components. This innovative architecture allows PSegNet to outperform established networks, like PointNet [Qi et al., 2017a], PointNet++ [Qi et al., 2017b] and PlantNet Li et al. [2022b] in both organ semantic and leaf instance segmentation tasks across tobacco, tomato, and sorghum plants.

More recently, Du et al. [2023] introduced PST, a method for segment siliques from rapeseed plants, excelling in both semantic and instance segmentation tasks (see its architecture in Figure 2.3). Inspired by dynamic voxelization [Zhou et al., 2020] and voxel feature encoding [Zhou and Tuzel, 2018], PST incorporated a dynamic voxel feature encoder that efficiently converts point-wise inputs into voxel-wise embeddings with learned features, thereby reducing information loss. Its self-attention module, adapted from the shifted-window self-attention [Liu et al., 2021; Fan et al., 2022], effectively captures the neighboring context for voxel feature learning. Additionally, PST integrated an instance segmentation head from PointGroup [Jiang et al., 2020], enhancing its instance segmentation capabilities. The results show that for both semantic and instance silique segmentation, PST outperformed PointNet++[Qi et al., 2017b], PAConv[Xu et al., 2021], and DGCNN [Zhang et al., 2021].

While these models have demonstrated significant performance improvements, training these segmentation models requires a large amount of annotated data to ensure optimal results. Additionally, their performance is susceptible to both the quantity and quality of the training dataset. Therefore, when deploying these models in plant phenotyping practices, a portion of the data still needs to be annotated manually after the data acquisition process, which is for model training or fine-tuning on a pre-trained model. Consequently, these approaches are remaining time-consuming and costly currently, primarily due to the need for manual labeling, which requires operators skilled in both plant biology and computer science.

2.2.4 Skeleton-based approaches

The skeleton is one of the simplified representations of a 3D point cloud model, which can intuitively reflect the morphology structure and topology structure of itself. Therefore, unlike clustering-based and model-fitting approaches, which primarily focus on the local geometric information of plant point clouds, skeleton-based methods can leverage both the local and global topological structure [Wang et al., 2023]. Given these characteristics, plant skeletons, which inherently encode the positions of plant organs, can be effectively used for plant organ segmentation tasks. To establish



(a) Sliced plant point cloud.

(b) Plant skeleton.

(c) Stem-leaf segmentation.

Figure 2.4: Slice clustering-based plant skeletonization and stem-leaf segmentation [Xiang et al., 2019].

skeleton-based plant organ segmentation, the first step is to generate a curve skeleton from the input plant point cloud.

One intuitive approach to extracting plant skeletons is based on slice clustering. Xiang et al. [2019] sliced an individual maize point cloud, which was reconstructed by a ToF camera, into several layers along the vertical axis (Fig. 2.4a). For each slice, the contained points were clustered into multiple clusters using Euclidean clustering, and the centroid of each cluster was denoted as a skeleton point. The skeleton points from adjacent layers were then connected to form an initial graph. By weighting the edges of this graph based on Euclidean distance, a maize skeleton was generated by computing a MST (as shown in Figure 2.4b). Finally, they applied an elliptical cylinder fitting algorithm to detect the stem and performed leaf instance segmentation by partitioning the skeleton graph at the junction nodes.

Although this skeletonization approach is simply, it assumes that the plant has an ideal vertical structure, thereby, horizontal parts of the plant are not well represented, because they tend to belong to a single cluster, resulting in only one skeleton point for the entire horizontal section. This significantly reduces the quality of the centeredness and the overall skeleton representation. Additionally, the curvature of the extracted skeleton is poor, often containing significant zigzag structures. These zigzag formations degrade the ability of the skeleton, and is hard to efficiently represent both the local and global geometric features of the original point cloud. Therefore, this approach has significant limitations.

The L_1-medial skeleton algorithm constructs the skeletal framework of a point cloud by iteratively computing local L_1 median points [Huang et al., 2013]. This process balances attractive and repulsive forces within a progressively expanding local neighborhood. Through this iterative contraction method, the algorithm efficiently identifies and extracts skeleton points, allowing for a robust representation of the underlying structure of the point cloud, even in the presence of noise, outliers, or incomplete data. Consequently, the L_1-medial skeleton algorithm has been utilized in plant skeletonization.

For example, Su et al. [2019] successfully used the L_1-medial skeleton algorithm to extract tree skeletons and accomplished the segmentation of photosynthetic and non-photosynthetic components of the tree based on that skeleton. Similarly, Ma et al. [2023] proposed a hierarchical segmentation method, which can be integrated with the L_1-medial skeleton algorithm to obtain satisfactory skeletons of rapeseed plants from point clouds. Based on the extracted discrete skeleton points, they employed the DBSCAN method to classify the skeleton points into several classes. A weighted unidirectional graph was then used to construct the connectivity among points in the same class. After a set of optimization operations, each silique and stem was assigned its own skeleton segments. By analyzing the length of these segments, they identified the silique class segments, thus achieving silique segmentation.

However, although L_1-medial skeleton algorithm could feasibly process cylindrical-shaped point clouds, adeptly extracting skeletons from objects like tree branches [Su et al., 2019] or rapeseed siliques [Ma et al., 2023], it is unsuitable for flat or planar structures, such as plant leaves. In these

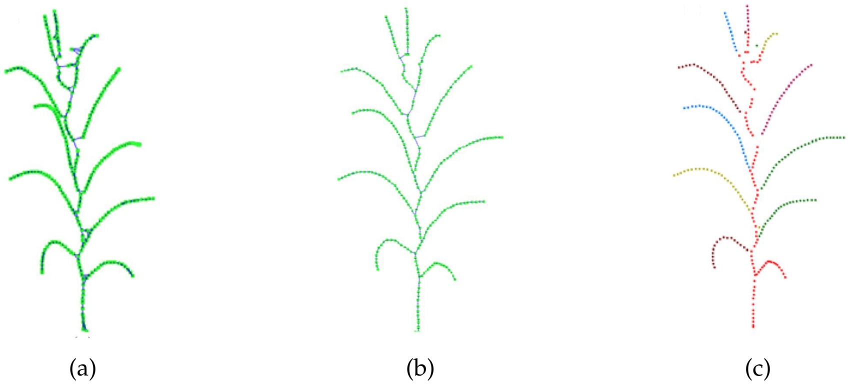


Figure 2.5: Plant skeleton connectivity optimization and decomposition [Wu et al., 2019]. (a) Initial skeleton connectivity. (b) Optimized skeleton connectivity. (c) Skeleton decomposition.

cases, the extracted leaf skeleton may not accurately represent the leaf veins and can sometimes fall outside the point cloud of leaf [Wu et al., 2019]. Thus, extracted plant skeletons from plants with leaves are unsatisfactory.

Moreover, inspired by Laplacian-based 3D mesh skeletonization [Au et al., 2008], Cao et al. [2010] proposed the LBC algorithm, which could process point clouds. This method iteratively contracts a point cloud using the cotangent-weighted Laplacian operator, which is constructed from the local one-ring Delaunay neighborhood. It has been proven highly effective in extracting curve skeletons from both cylindrical objects [Li et al., 2020] and surfaces with boundaries, such as plant stems and leaves, and has been widely adapted for plant skeleton extraction.

For instance, Wu et al. [2019] applied this algorithm to contract the point cloud of maize into a line-like structure. After applying adaptive sampling, which was weighted by the directionality degree of each point, to extract key skeleton points that could closely approximate the original shape, they established connectivity among the points under the constraint of maize geometric characteristics (Fig. 2.5a and 2.5b). This process ensured that the curve skeleton had a tree-like structure and preserved the accurate topology of maize. By breaking the junction nodes, they could segment the curve skeleton into stem and leaf segments, thereby achieving stem-leaf segmentation of the maize plant (as shown in Figure 2.5c).

Miao et al. [2021] also applied the LBC algorithm to contract a maize point cloud and established stem-leaf segmentation by decomposing the resulting curve skeleton. However, during the contraction iterations, the algorithm tends to ignore small parts and merge them with nearby branches. Therefore, the quality of the final curve skeleton and the segmentation result directly obtained from this curve skeleton can be unsatisfactory. To address this issue, Miao et al. [2021] further refined the segmentation results by using a distance-based strategy.

Furthermore, Wang et al. [2024] utilized the LBC algorithm to obtain a curve skeleton of a tomato plant by connecting each extracted skeleton point to its two neighbors. Then, they detected the main stem of the tomato by finding the shortest path from the root vertex to the highest vertex along the growth direction. After identifying and removing the main stem, they achieved leaf instance segmentation by applying supervoxel clustering based on Euclidean distance.

Although the LBC algorithm can effectively contract plant stems and leaves into line-like objects, it still has significant drawbacks. As seen in Figure 2.5a, the discrete skeleton points contracted by LBC algorithm do not generally approximate the original shape of the input plant point cloud. The skeleton contains zigzag structures that fall outside the point cloud, failing to satisfy the centeredness criterion. Moreover, the integrity of the skeletons also cannot be guaranteed, and abnormal branches often appear. Applying such skeletons directly to plant segmentation tasks can reduce efficiency and accuracy.

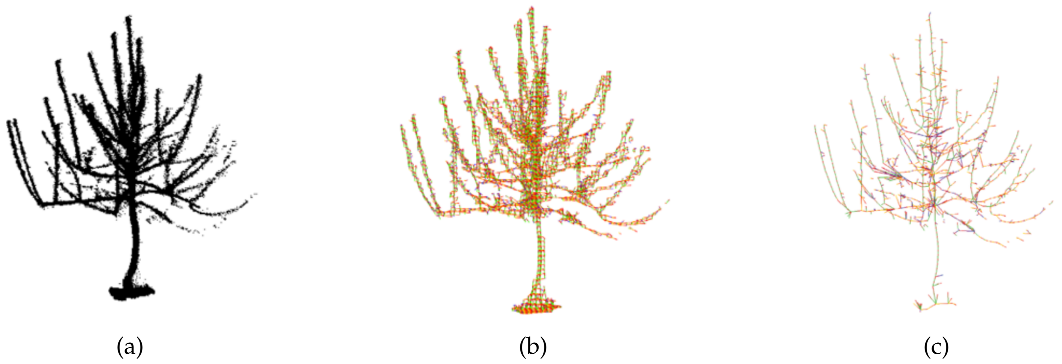


Figure 2.6: Extracted tree skeleton by using SkelTre [Bucksch et al., 2010]. (a) Original tree point cloud. (b) Extracted octree graph. (c) Extracted tree skeleton.

Therefore, to obtain a skeleton that is both topologically and geometrically accurate, maintains integrity, has precise localization, and preserves correct branch connectivity, interactive corrections or geometric constraints must be incorporated into the algorithm to ensure high-quality results.

2.3 Plant Skeleton Extraction

The application of plant curve skeletonization extends beyond plant organ segmentation; it can also be utilized to extract various plant canopy morphological traits (e.g., branch inclination angles, stem length, leaf length, and stem diameter) or to improve the quality of 3D plant model reconstruction. Therefore, in this section, we broaden the discussion to include plant skeletonization algorithms that are not limited to the scope of organ segmentation. We present several studies that incorporate plant skeletonization as a component of their proposed pipelines. However, we exclude pipelines that use the L_1-medial skeleton or LBC algorithm, as these have already been discussed in Section 2.2.4.

An early work introduced by Bucksch et al. [2010] proposed SkelTre, which could efficiently extract the tree skeleton from the point cloud. In their pipeline, the skeleton points were extracted based on an octree built upon the point clouds. Neighborhood information of octree cells was used to generate an octree graph (Fig. 2.6b). The vertices of the octree graph are positioned at the center of gravity of the points within each cell and are connected by edges when two cells share adjacent faces. The octree graph is subsequently simplified into a skeleton through a graph reduction process. (Fig. 2.6c). This octree-based approach could efficiently preserve both topology and surface information and maintain the centeredness of the extracted skeleton.

Du et al. [2019] introduced AdTree, a novel method for reconstructing accurate 3D tree models from point cloud. Among their reconstruction pipeline, they proposed an efficient approach to extract skeleton from tree point cloud, which ensures both geometric and topological accuracy in the final tree model. Their approach begins by constructing an initial skeleton by computing an MST, which is computed from an initial graph generated through Delaunay triangulation of the input point cloud, and the edges of this initial graph are weighted based on Euclidean distances between points. Then, a simplification process is employed to refine the initial skeleton to obtain the final tree skeleton, which involves removing small, noisy branches and merging vertices and edges with similar properties (Fig. 2.7b).

Chaudhury and Godin [2020] introduced a stochastic optimization framework aimed at refining the initial coarse skeleton generated by the skeletonization algorithm proposed by Xu et al. [2007]. This framework leverages the input point cloud data to preserve biological relevance throughout the optimization process. By iteratively manipulate the skeleton points using a Gaussian Mixture model and Expectation Maximization algorithm, the framework corrects inaccuracies in the initial coarse skeleton, and aligns the skeleton with the original geometry of plant, thereby, ensuring a more uniform distribution of final skeleton points. As a result, the refined skeleton avoids the zigzag

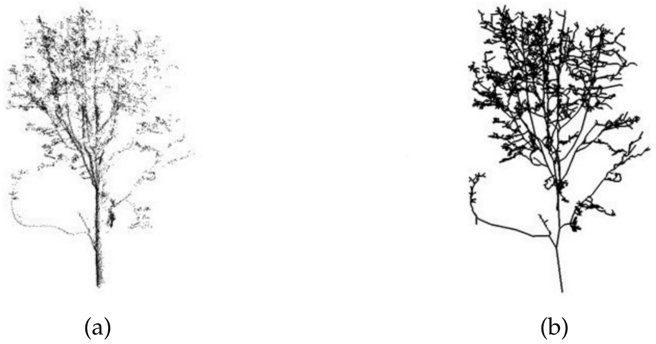


Figure 2.7: Extracted tree skeleton by using AdTree [Du et al., 2019]. (a) Original tree point cloud. (b) Extracted tree skeleton.

structures and better captures fine structural details compared to the initial coarse skeleton extracted by the method introduced by Xu et al. [2007].

To extract tree inclination angles from point clouds, Wu et al. [2021] proposed the WoodSKE method, which is designed to efficiently extract a set of discrete skeleton points to approximate tree skeletons while preserving structural integrity and centeredness. This method first contracts the original point cloud into a set of coarse skeleton points by analyzing the local neighborhood information of each point. It then refines the skeleton points through a thinning process, further adjusting skeleton points based on their closest Euclidean distance to the original point cloud. WoodSKE effectively maintains the topological and geometrical features of tree branches with complex structures and outperforms L_1-medial skeleton and LBC algorithm.

Although the aforementioned skeletonization algorithms can efficiently extract the skeletons of tree branches (long, narrow shapes), they do not adequately address the skeletonization of plant leaves. Skeletons extracted from trees with leaves often suffer from poor quality [Bucksch et al., 2010], as existing frameworks typically focus on plants without leaves [Chaudhury and Godin, 2020]. Therefore, incorporating plant leaves into the skeletonization process is worth exploring to better reconstruct the geometry of leafy plants (e.g., herbaceous plants).

3 Methodology

Based on the research objectives and questions defined in Chapter 1 and the review of plant skeletonization methods in Chapter 2, we select the LBC algorithm as the baseline technique for extracting plant skeletons in this study because of its proficiency in handling objects with horizontal, cylindrical, and flat geometries. While the LBC algorithm shows a potential in extracting curve skeletons from leafy plant point clouds, further refinements are still necessary to achieve a precise curve skeleton, which is the basis of high quality plant organ segmentation results.

This chapter presents our methodology, which includes geometric contractions and calibration operations to refine LBC algorithm and obtain satisfactory plant curve skeletons. We also outline the subsequent process for establishing stem-leaf segmentation based on the extracted skeleton. We aim to (a) construct an accurate curve skeleton of ***P. lapathifolium***, and (b) achieve precise segmentation of its stem and leaves. Eventually, we aim to introduce a novel skeleton-based plant organ segmentation pipeline for leafy plants that can be applied in real-world production scenarios.

This chapter is organized as follows:

- Section 3.1 provides an overview of our proposed framework.
- Section 3.2 introduces refinement strategies for the LBC algorithm, enhancing its ability to contract a set of discrete skeleton points from point clouds, which could approximate the original plant shape accurately in both topology and geometry.
- Section 3.3 presents a calibration operation to ensure the centeredness of skeleton points, along with a brief description of the graph-building process.
- Section 3.4 discusses a graph-based approach for stem-leaf segmentation built upon the skeleton graph.
- Section 3.5 briefly outlines the pipeline for reconstructing our testing leafy plant point clouds.
- Section 3.6 details the evaluation criteria for the stem-leaf segmentation process.

3.1 Overview

Overall, our framework takes an individual plant point cloud as input. The plant point cloud should already be denoised and segmented from the background and contain only the plant stem and leaves, as denoising, background, and individual plant segmentation fall outside the scope of this study. The framework produces two outputs: 1) the plant curve skeleton and 2) segmented point cloud clusters in two semantic classes: stem and leaf. Additionally, during segmentation, each point will be assigned stem-leaf semantic labels as well as individual leaf instance labels.

Figure 3.1 provides an overview of the module design for the proposed methodology. Our proposed framework consists of three main steps: 1) Point Cloud Contraction, 2) Skeleton Graph Generation, and 3) Stem-leaf Segmentation.

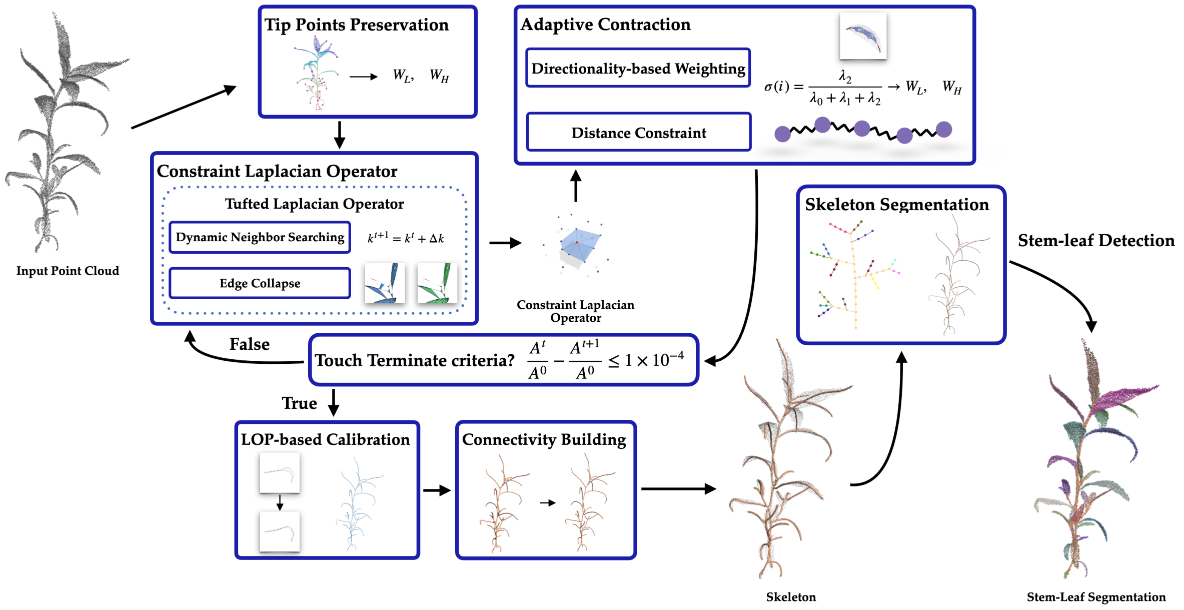

Tip Points Preservation
$\rightarrow W_L, \; W_H$
Adaptive Contraction
Directionality-based Weighting
$\sigma(i) = \frac{\lambda_2}{\lambda_0 + \lambda_1 + \lambda_2} \rightarrow W_L, \; W_H$
Distance Constraint
Input Point Cloud
Constraint Laplacian Operator
Tufted Laplacian Operator
Dynamic Neighbor Searching
$k^{t+1} = k^t + \Delta k$
Edge Collapse
Constraint Laplacian Operator
Skeleton Segmentation
Stem-leaf Detection
False
Touch Terminate criteria? $\frac{A^t}{A^0} - \frac{A^{t+1}}{A^0} \leq 1 \times 10^{-4}$
True
LOP-based Calibration
Connectivity Building
Skeleton
Stem-Leaf Segmentation

Figure 3.1: An overview of the proposed framework. **Point Cloud Contraction**: Constraint Laplacian Operator, Adaptive Contraction and Tips Points Preservation. **Skeleton Graph Generation**: LOP-Based Calibration and Connectivity Building. **Stem-Leaf Segmentation**: Skeleton Segmentation and Stem-Leaf Detection.

Point Cloud Contraction. The point cloud shrinking step is generally based on the LBC algorithm. It takes the individual plant point cloud as input and outputs a zero-volume point set with the same number of points. During the contraction process, we enhance the Tufted Laplacian operator proposed by Sharp and Crane [2020] with additional **Dynamic Neighbor Searching** and **Edge Collapse** modules to form the **Constraint Laplacian Operator**. Additionally, we introduce two other modules, **Adaptive Constraint** and **Tip Points Preservation**, to further improve the skeleton quality.

Skeleton Graph Building. The output of the previous step is a set of discrete skeleton points that could approximate the plant curve skeleton. As the points would have slightly shifted away from the original point cloud during the contraction process, we modified the LOP operator to calibrate the downsampled skeleton points. Next, we triangulate the calibrated skeleton points by connecting neighbor points and compute an MST to construct the initial skeleton graph. To minimize segmentation errors, we prune the skeleton graph by removing noisy branches.

Stem-Leaf Segmentation. With the generated skeleton graph, we first segment the extracted skeleton graph by iteratively traversing it from the leaf nodes to their nearest junction nodes. We then assign point-wise semantic and instance labels using the nearest projection strategy according to the segmented skeleton segments.

3.2 Point Cloud Contraction

3.2.1 Overview of Laplacian-based skeleton extraction

The Laplacian-based skeletonization technique was first introduced by Au et al. [2008] for contracting 3D closed meshes and was later adapted for point cloud skeletonization by Cao et al. [2010]. The contraction process involves removing details and noise by applying Laplacian smoothing, which moves the vertices along their approximate mean curvature normal directions. Using the cotangent-weighted Laplacian operator L, the mean curvature normals can be approximated by the cotangent-weighted Laplacian coordinates, given by $\delta_i = -4A_i\kappa_i\mathbf{n}_i$, and $\boldsymbol{\delta} = L\mathbf{P} = [\delta_1^\top, \delta_2^\top, \ldots, \delta_n^\top]^\top$ [Desbrun et al., 1999]. Here, A_i represents the local one-ring area of vertex p_i, κ_i denotes the approximate local mean curvature normal at vertex p_i, $\mathbf{n}_i$ is the approximate surface normal of vertex p_i, and $\mathbf{P}$ represents the vertex positions. By solving the Laplacian equation $L\mathbf{P} = 0$, the local one-ring areas are minimized, leading to the minimization of the overall volume of the object.

In this context, LBC algorithm can be formulated as an optimization problem (Eq. 3.1) that directs the movement of points by balancing the contraction weights $\mathbf{W}_L$ and the attraction weights $\mathbf{W}_H$ [Au et al., 2008].

$$\begin{bmatrix} \mathbf{W}_L L \\ \mathbf{W}_H \end{bmatrix} \mathbf{P}^{t+1} = \begin{bmatrix} 0 \\ \mathbf{W}_H \mathbf{P}^t \end{bmatrix} \tag{3.1}$$

By iteratively solving this linear system, the contraction forces guide the points to move vertically toward the mean curvature normals under the constraints from the attraction forces, gradually shrinking the object to zero volume while preserving the critical characteristics of the original geometry.

We can find that the Laplacian-based skeletonization technique relies on computing the cotangent-weighted Laplacian operator L, making the connectivity between points a crucial factor. To adapt this method for point clouds, Cao et al. [2010] proposed a technique to compute L without requiring explicit topology. This approach constructs a local planar Delaunay triangulation for each point and its k nearest neighbors, which are projected onto an estimated tangent plane. By determining the connectivity of one-ring neighbors through this triangulation, the L can be derived.

LBC algorithm is a geometric skeletonization method that takes a 3D point cloud $\mathbf{P} \in \mathbb{R}^{n\times 3}$ as input and performs iterative contraction. During each iteration, the linear system (Eq. 3.1) is solved by minimizing the quadratic energy (Eq. 3.2) to update the point cloud, generating $\mathbf{P}^{t+1}$.

$$\|\mathbf{W}_L^t \mathbf{L}^t \mathbf{P}^{t+1}\|^2 + \sum_i \mathbf{W}_{H,i}^2 \|p_i^{t+1} - p_i^t\|^2 \tag{3.2}$$

The weighting matrices $\mathbf{W}_L$ and $\mathbf{W}_H$ are updated according to Equation 3.3, where A_i^t and A_i^0 represent the current and initial neighborhood extents of point p_i.

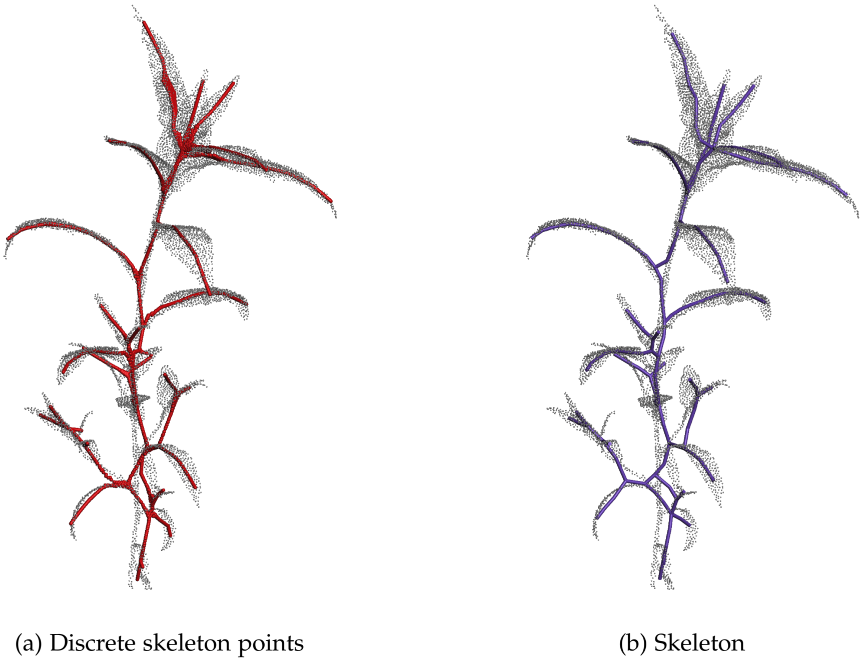
$$\mathbf{W}_L^{t+1} = s_L \mathbf{W}_L^t, \quad \mathbf{W}_{H,i}^{t+1} = \mathbf{W}_{H,i}^0 \frac{A_i^0}{A_i^t} \tag{3.3}$$

In each iteration, the L^{t+1} is reconstructed by using $\mathbf{P}^{t+1}$ and the connectivity information obtained at the first iteration. The iterative process continues until a zero-volume point set is reached based on a user-defined criterion. For example, this criterion could compare the ratio of the mean value of A^{t+1} to the mean value of A^0, with the ratio of the mean value of A^t to the mean value of A^0 [Au et al., 2008; Cao et al., 2010].

Figure 3.2 illustrates the result of the LBC algorithm on our captured plant point cloud. In the LBC algorithm, the skeleton (Fig. 3.2b) is computed based on contracted discrete skeleton points through a set of graph-based operations. Therefore, the more accurately the discrete skeleton points approximate the actual skeleton shape, the more accurate the final skeleton will be.

Overall, the LBC algorithm successfully contracts this plant point cloud to a nearly curve skeleton shape, while several drawbacks exist (Fig. 3.3).

There are several parts still under-contraction. The curve skeleton approximated by those contracted skeleton points do not fit the original shape well, especially on the junction part between the main stem and the lateral stem or leaves, and exists zigzag structures. Moreover, some tiny parts are been eliminated due to the over-contraction. Therefore, our proposed **Point Cloud Contraction** step will focus on addressing these limitations and eliminating their negative effects.



(a) Discrete skeleton points

(b) Skeleton

Figure 3.2: Discrete skeleton points and skeleton computed by LBC algorithm. Gray points represent the original point cloud; Red points indicate the contracted skeleton points; Purple cylinder present the skeleton. All the hyperparameters settings follow the recommended instructions in [Cao et al., 2010] and their MATLAB implementation.

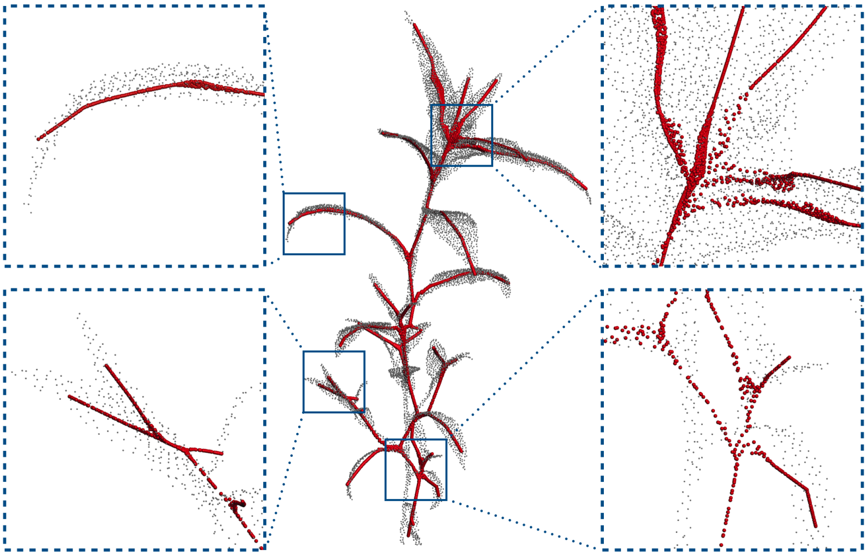
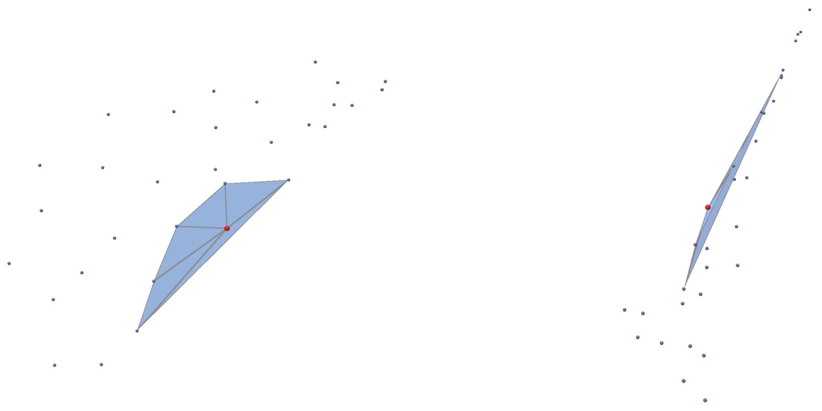


Figure 3.3: Skeleton points contracted by LBC algorithm [Cao et al., 2010].



(a)

(b)

Figure 3.4: Diagram of changes of one-ring neighborhood during iterations. (a) On the initial point cloud. (b) On the point cloud after two contraction iterations.

As illustrated in Figure 3.1, during each contraction iteration, a constraint Laplacian operator is computed. This primarily involves calculating a Tufted Laplacian operator of the point cloud [Sharp and Crane, 2020]. The computation process is enhanced by our additional **Dynamic Neighbor Searching** and **Edge Collapse** modules, as detailed in Section 3.2.2. After obtaining the constraint Laplacian operator $\mathcal{L}$ (to distinguish with ordinary cotangent-weighted Laplacian operator, hereafter, we use $\mathcal{L}$ to represent our constraint Laplacian operator), the point cloud begins to be contracted under the constraints of our adaptive contraction module, which is guided by Directionality-based weighting and Distance Constraint, as discussed in Section 3.2.3. To ensure the global geometry of the plant point cloud is preserved as accurately as possible, we introduce Tip Points Preservation in Section 3.2.4. These improvements result in a refined set of skeleton points that could approximate the original point cloud shape well and be ready for subsequent processing.

3.2.2 Constrained Laplacian operator

Tufted Laplacian Operator

As we mentioned in Section 3.2.1, Cao et al. [2010] applied strategy, which only computed the one-ring Delaunay connectivity information for each point only during the first iteration, and the L is then updated from this initial connectivity and the contracted point coordinates. This approach has inherent limitations. As the contraction progresses, points tend to be shrunk into a line-like structure, causing the point distribution to become sharp and irregular. Thus, the triangulation constructed based on the initial connectivity information becomes increasingly incompatible with the intrinsic Delaunay criteria (Fig. 3.4).

This incompatibility leads to the formation of skinny triangles, which may result in non-manifold triangulation and worsen the conditioning of the Laplacian. Consequently, this can introduce negative edge weights and corresponding negative cotangent weights in the L These irregular weights can lead to abnormal contraction directions, which, in turn, contribute to the zigzag artifacts and abnormal branches.

Therefore, we adapt the novel and robust strategy proposed by Sharp and Crane [2020] for constructing L, which is capable of deriving a satisfactory and robust L (referred to as the Tufted Laplacian operator) from highly irregular and non-manifold triangle meshes. We apply this strategy in the LBC process to derive the Tufted Laplacian operator from the point cloud during each iteration. With a union of triangles made from all the local triangles of each point in the point cloud, Sharp and Crane [2020] employ the tufted cover and flipping operations to generate intrinsic Delaunay triangulation,

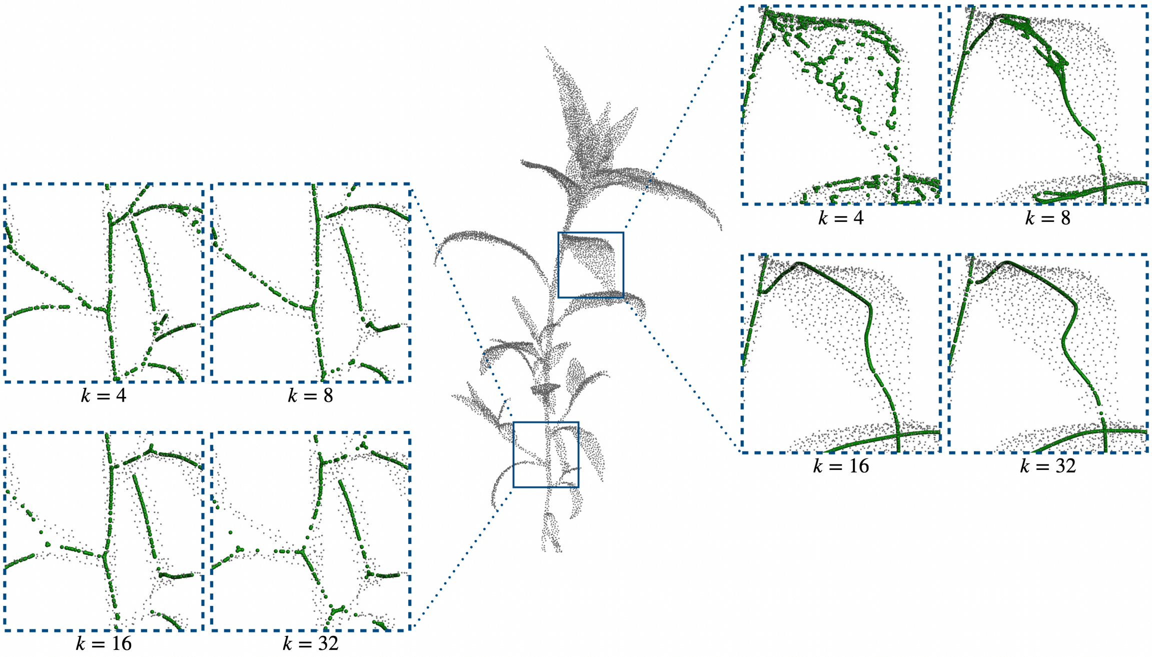


Figure 3.5: Effect of different fixed k values on the contracted skeleton points. In this experiment, we replace L with the Tufted Laplacian operator and heuristically set the initial value of $\mathbf{W}_L$ to 1.0, which is also applied in subsequent experiments.

thereby deriving Tufted Laplacian operator from the input point cloud. With this Tufted Laplacian operator for the point cloud, we could ensure the contraction direction during the iterations.

In this study, we improve the Tufted Laplacian operator construction strategy by integrating **Dynamic Neighbor Searching** and **Edge Collapse** modules to preserve important small-scale features during the contraction process. The two modules aim to address this from two perspectives: first, by adjusting the neighborhood size k used for local triangulation construction, and second, by applying targeted manipulations to the union of all constructed local triangles.

Dynamic Neighbors Searching

The contraction distance and direction of a point during the contraction iteration are determined by the values in its corresponding row of the L, which is related to its one-ring neighborhood. The size of k influences the one-ring neighborhood by directly affecting the initial triangulation region, which is formed by the point and its k-nearest neighbors. Although the one-ring neighborhood is generated after the initial triangulation through tufted overs and flipping operations, the k value and the average size of the one-ring neighborhood remain positively correlated under certain thresholds. Regardless that a more significant k value generally provides a more accurate representation of local geometric structures [Belkin et al., 2008; Hildebrandt and Polthier, 2011], however, such L can also smooth out finer geometric details at more minor scales.

Therefore, as the value of k increases, more local features (e.g., newly emerging leaves or small branches) in the point cloud tend to be ignored in the final skeleton points. Additionally, the zigzag artifacts in the junction regions become increasingly prominent (as shown in Figure 3.5, *left*). However, if k is too small, many abnormal skeleton points are extracted, particularly on the plant leaves, which degrades the quality of the final skeleton (as shown in Figure 3.5, *right*).

This phenomenon is primarily due to the increasing density of the point cloud during the iterations. At lower k values and higher point cloud densities, very fine local geometric features are captured, resulting in the extraction of abnormal skeleton points.

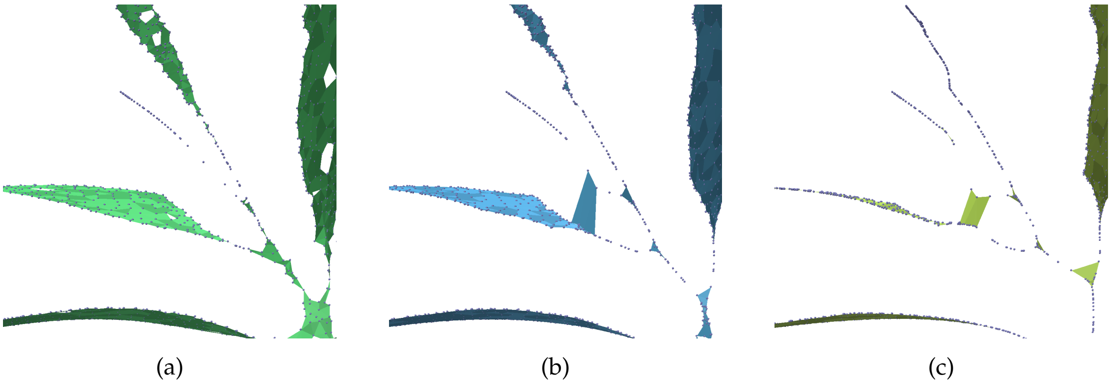


Figure 3.6: Diagram of one-ring neighborhood changes with contraction iterations and raising k. (a) Iteration 1, $k^1 = 8$. (b) Iteration 2, $k^2 = 12$. (c) Iteration 3, $k^3 = 16$.

To address the trade-off in selecting the optimal k value, we propose **Dynamic Neighbors Searching**, which dynamically adjusts the k value during each contraction iteration. This approach balances the preservation of fine details with the prevention of abnormal skeleton point extraction. **Dynamic Neighbors Searching** allows the k value to increase iteratively during the contraction process, as shown in Equation 3.4:

$$k^{t+1} = \begin{cases} k^t & k^t + \Delta k > k_{\max} \\ k^t + \Delta k & \text{otherwise} \end{cases}, \tag{3.4}$$

where k^{t+1} is the k value used for building the initial union triangles for Tufted Laplacian operator contraction at iteration $t+1$. The parameter Δk determines the increment of k in each iteration, while $k_{\max}$ serves as the upper limit to prevent excessively large values.

This dynamic adjustment preserves fine geometric details in the initial contraction phases while preventing the formation of abnormal skeleton points and noisy local branches in subsequent iterations.

Edge Collapse

While **Dynamic Neighbor Searching** effectively maintains the balance between preserving local geometric features and avoiding abnormal skeleton points, it may not always be sufficient to preserve small features throughout the entire contraction process fully. As k increases, the one-ring neighborhood of points on junction regions or closely spaced leaves would expand, potentially incorporating unwanted neighbors from other stems or leaves (Fig. 3.6). This expansion can lead to unintended offsets in skeleton points, which should ideally remain at junction nodes, and tiny leaf emission.

To address this issue, in addition to setting a lower initial k value to better retain local geometric features, we propose directly collapsing long triangle edges within the initial union of all local triangles. This method directly manipulates the connectivity between points before further processing, ensuring a more precise contraction of the intrinsic Delaunay triangulation.

Assuming a uniformly distributed input point cloud, we identify "outlier edges", defined as those whose lengths are more than three standard deviations above the mean edge length within the initial union of all local triangles. The minimum value among these outliers is set as the threshold for edge collapse operation. We then iterate through all edges in the initial union of all local triangles, collapsing those that exceed the threshold. In cases where a point becomes isolated after an edge collapse, we reconnect it to its two nearest neighbors to prevent the formation of isolated points.

By integrating the above two modules into the Tufted Laplacian operator process, we can obtain a constrained Laplacian operator $\mathcal{L}$ that efficiently maintains local geometric features while preventing the formation of noisy local branches or abnormal skeleton points.

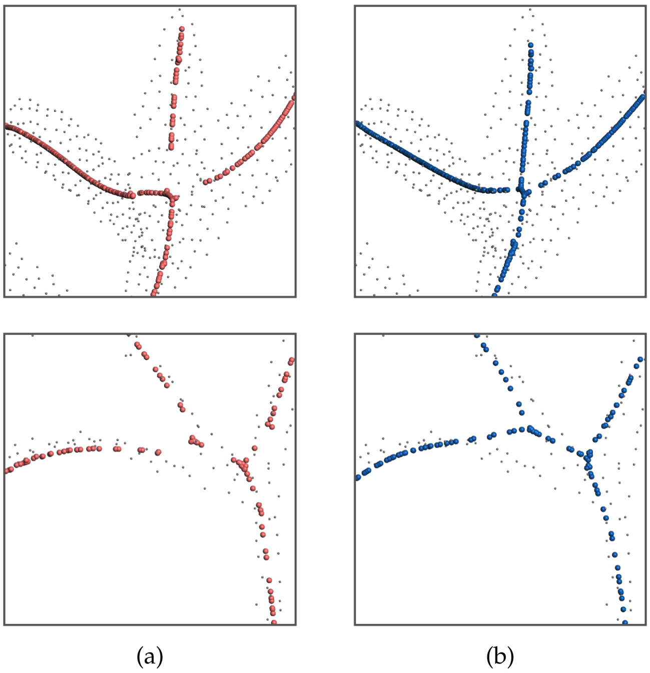


Figure 3.7: Gaps between the skeleton points and the performance of Distance Constraint. In this experiment, we replace the Tufted Laplacian operator with the constraint Laplacian operator $\mathcal{L}$, which is also applied in all subsequent experiments. (a) Gaps. (b) Maintained by Distance Constraint.

3.2.3 Adaptive contraction

In the LBC algorithm, the contraction and attraction forces of a point are determined by changes in the extent of its one-ring neighborhood, which indicates the current degree of contraction. However, when using the $\mathcal{L}$, the number of one-ring neighbors changes due to **Dynamic Neighbor Searching** and **Edge Collapse** modules, making the relationship between the neighborhood extent and the contraction degree less explicit. This affects the correct calculation of contraction and attraction forces and can result in contraction progress or abnormal skeleton points.

Additionally, it is important to note that as we actively manipulate the one-ring neighborhood, we compute the dual vertex area for each point on the intrinsic Delaunay triangulation, which is constructed with k value fixed at 30, and use this area to represent the local one-ring area of each point. The termination criterion is that the mean area size change ratio between two consecutive iterations should not be less than 1×10^{-4}. This is applied in all subsequent experiments.

While the $\mathcal{L}$ effectively preserves small details by adjusting the size of the one-ring neighborhoods, it also has notable drawbacks. It may ignore essential connectivity when manipulating the size of the one-ring neighborhoods, potentially leading to gaps in the approximated curve skeleton (Fig. 3.7a) and introducing noise or incorrect edge connections in the subsequent skeleton graph building steps.

To overcome these drawbacks while keeping the advantages of the $\mathcal{L}$, we propose an adaptive contraction strategy to explicitly manage the contraction and attraction forces during the contraction process. An intuitive approach is to use an alternative indicator to represent the contraction degree of each point and apply different weighting matrices to control the appropriate contraction and attraction forces. Additionally, to address gaps between skeleton points that should be adjacent, we introduce **Distance Constraint** that regulates the maximum allowable distance between a point and its neighbors during contraction.

Directionality-based Weighting

The directionality degree σ, introduced by Huang et al. [2013], is utilized to evaluate the contraction status of different regions. This is calculated by the weighted eigenvalue ratio of each point from the weighted covariance matrix,

$$\begin{aligned} \mathbf{cov}_i &= \sum_{i' \in I \backslash \{i\}} \theta\left(\|p_i - p_{i'}\|\right)\left(p_i - p_{i'}\right)^\top \left(p_i - p_{i'}\right) \\ \theta(x) &= e^{\frac{-x^2}{(h/2)^2}}, \end{aligned} \tag{3.5}$$

where h represents the radius of the search sphere used to identify neighboring points for p_i.

$$\sigma_i = \frac{\lambda_i^2}{\lambda_i^0 + \lambda_i^1 + \lambda_i^2}, \quad \text{where } \lambda_i^0 \leq \lambda_i^1 \leq \lambda_i^2 \tag{3.6}$$

Moreover, the work by Huang et al. [2013] and Wang et al. [2017] has demonstrated that smoothing σ_i of point p_i by averaging the σ values among its neighbors enhances robustness to noise and improves the capacity of indicating contraction status (see in Figure 3.8). Inspired by those works, we introduce a constraint schema for updating the weighting matrices based on the smoothed σ. After each iteration, the update equations for the weighting matrices $\mathbf{W}_L$ and $\mathbf{W}_H$ are redesigned as follows:

$$\mathbf{W}_{L,i}^{t+1} = \begin{cases} 0 & \text{smooth\_}\sigma_i^t \geq \epsilon \\ \mathbf{W}_{L,i}^t & \epsilon > \text{smooth\_}\sigma_i^t \geq \epsilon - 0.1 \\ s_L \mathbf{W}_{L,i}^t & \text{otherwise} \end{cases} \quad \mathbf{W}_{H,i}^{t+1} = \begin{cases} \phi & \text{smooth\_}\sigma_i^t \geq \epsilon \\ \mathbf{W}_{H,i}^t & \text{otherwise} \end{cases} \tag{3.7}$$

Here, ϵ represents the threshold for the smooth_σ that distinguishes well-contracted points from other points, and ϕ is a constant that controls the attraction weight.

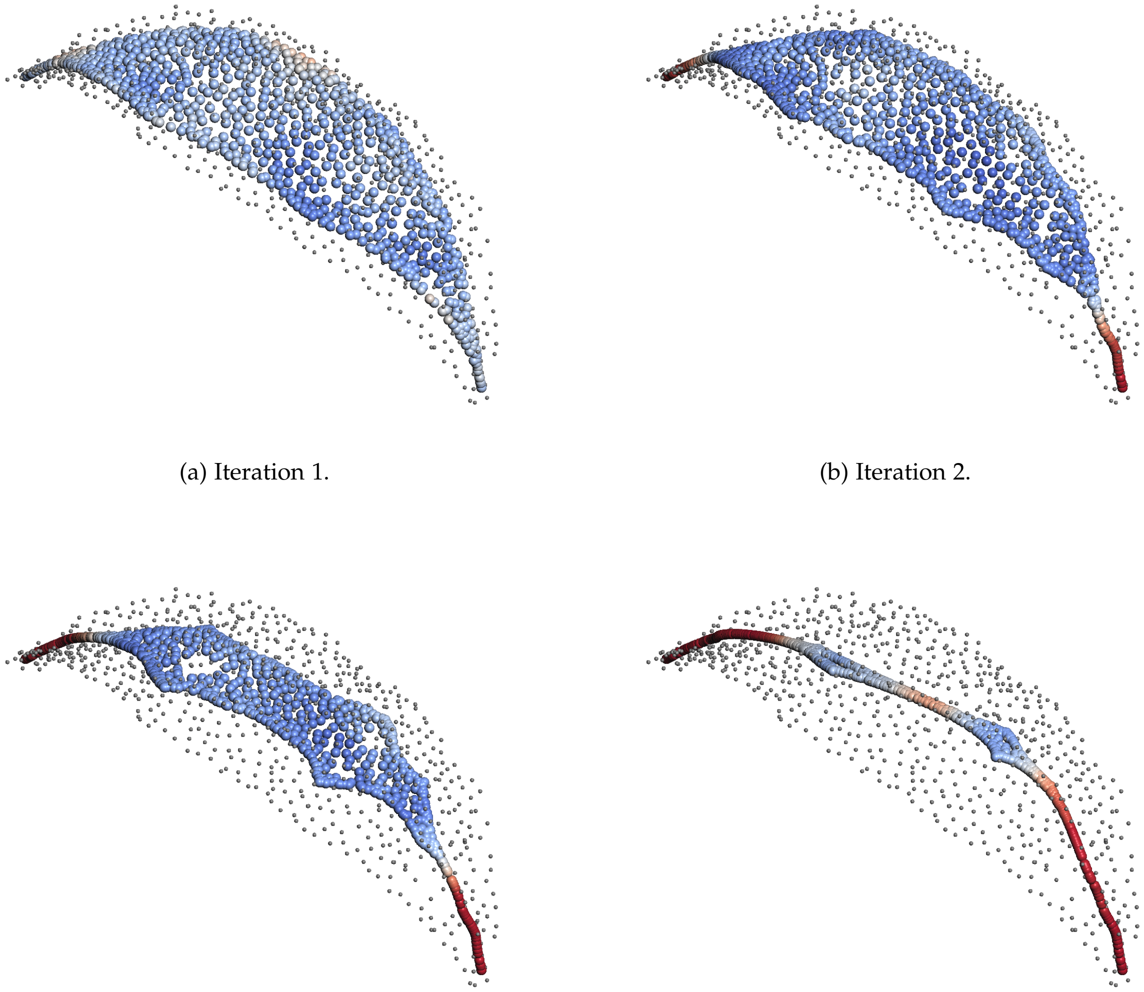
In accordance with the positive correlation between contraction force and the ratio of $\mathbf{W}_L/\mathbf{W}_H$, where a larger $\mathbf{W}_L/\mathbf{W}_H$ ratio signifies stronger contraction forces, we reformulate the updated equations for $\mathbf{W}_L$ and $\mathbf{W}_H$ (Eq. 3.7). Moreover, to prevent over-contraction caused by an excessively large $\mathbf{W}_L^i/\mathbf{W}_H^i$ ratio, we heuristically set a maximum value for $\mathbf{W}_L^i$ at 9.0. These updated equations are designed to dynamically enhance contraction forces in regions of severe under-contraction, stabilizing forces in regions with slight under-contraction, and apply a soft constraint to maintain equilibrium in well-contracted regions.

Distance Constraint

Building upon the efficient method for preserving edge lengths in 2D mesh deformation by Weng et al. [2006], we adapt their energy formulation to penalize changes in edge lengths, enforcing a distance constraint between points and their one-ring neighbors during contraction (Eq. 3.8).

$$\begin{aligned} &\sum_{(i,j) \in \text{Edge}_\mathcal{T}} \left\|(p_i - p_j) - e(p_i, p_j)\right\|^2 \\ &e(p_i, p_j) = \begin{cases} (p_i - p_j) & \left\|p_i - p_j\right\|^2 \leq d_{\max} \\ \frac{d_{\max}}{\left\|p_i - p_j\right\|^2}(p_i - p_j) & \left\|p_i - p_j\right\|^2 > d_{\max} \end{cases}, \end{aligned} \tag{3.8}$$

where Edge$_\mathcal{T}$ represents a set of edges of "connected points", derived from the upper/lower triangular portion of the Tufted Laplacian operator, as computed using the strategy proposed by Sharp and Crane [2020]. The term $d_{\max}$ refers to the user-defined maximum allowable distance between points. In this study, we fixed the k value at 4 to identify one-ring neighbors for each point for constructing Edge$_\mathcal{T}$. A larger k value or dynamic k value will make it difficult to determine a suitable value for $d_{\max}$.



(a) Iteration 1.

(b) Iteration 2.

(c) Iteration 3.

(d) Iteration 4.

Figure 3.8: Changes of smooth_σ value of each point with contraction iterations. The red points indicate the higher smooth_σ value.

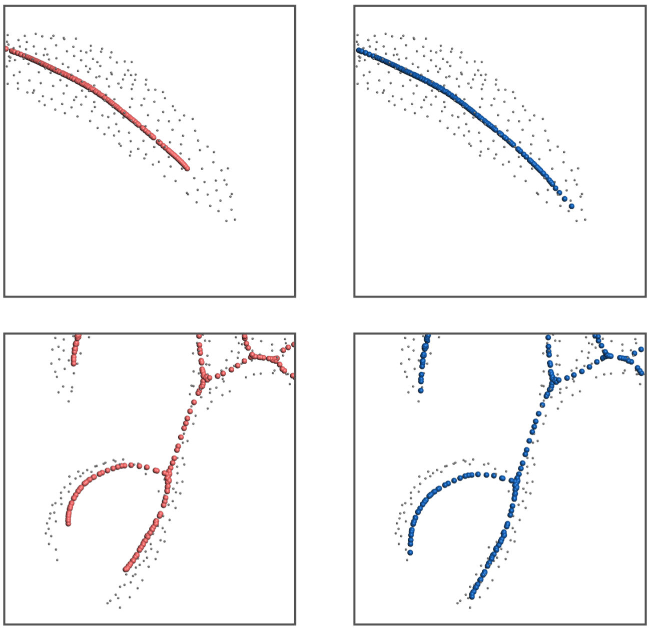

(a) (b)

Figure 3.9: Offsets of skeleton points in the tip regions and the performance of Tip Points Preservation. In this experiment, all the above modules are enabled. (a) Offsets. (b) Maintained by Tip Points Preservation.

By transforming this constraint into a matrix form, we can integrate it into Equation 3.2. Consequently, the updated positions of the contracted points are computed by minimizing the sum of all energy terms:

$$\|\mathbf{W}_L^t \mathbf{L}^t \mathbf{P}^{t+1}\|^2 + \sum_i \mathbf{W}_{H,i}^2 \|p_i^{t+1} - p_i^t\|^2 + \|\mathbf{W}_D \mathbf{H} \mathbf{P}^t - \mathbf{W}_D e(\mathbf{P}^t)\|^2 \tag{3.9}$$

Thereby, the previous optimization problem (Eq. 3.1) can now be reformulated as:

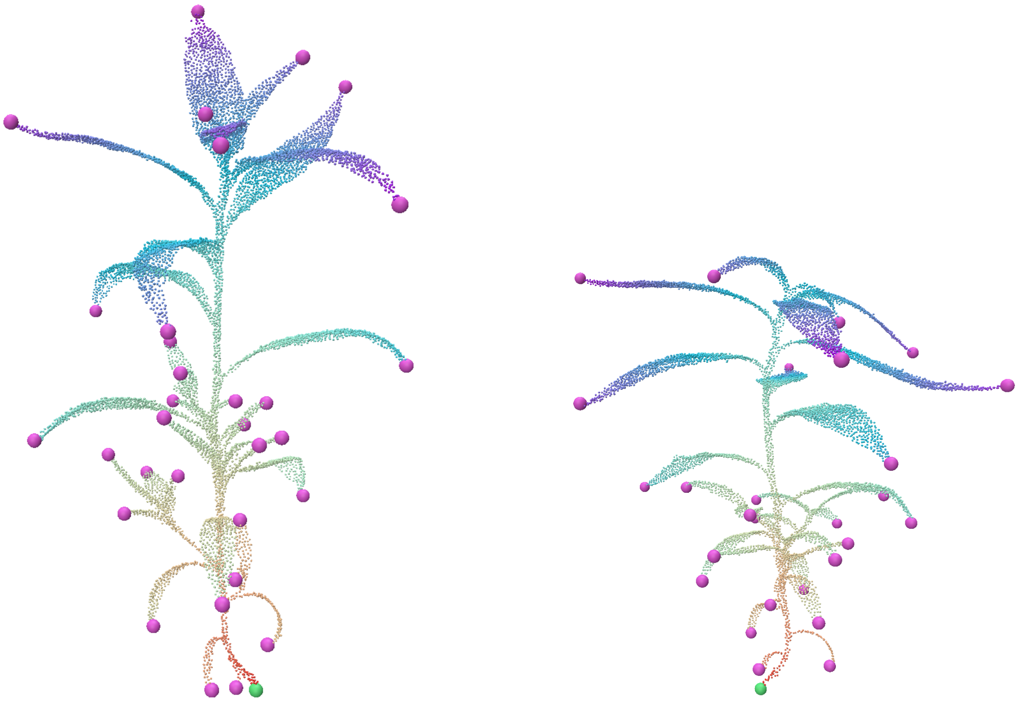
$$\begin{bmatrix} \mathbf{W}_L L \\ \mathbf{W}_H \\ \mathbf{W}_D \mathbf{H}^t \end{bmatrix} \mathbf{P}^{t+1} = \begin{bmatrix} 0 \\ \mathbf{W}_H \mathbf{P}^t \\ \mathbf{W}_D e(\mathbf{P}^t) \end{bmatrix}, \tag{3.10}$$

where $\mathbf{W}_D$ is a constant weight associated with the distance constraint (in this study, we find it works well when it is set to 9.0), $\mathbf{H}$ is a matrix of size $\text{rows}(\text{Edge}_{\mathcal{T}}) \times \text{rows}(\mathbf{P})$, and $e(\mathbf{P}^t)$ is a matrix of size $\text{rows}(\text{Edge}_{\mathcal{T}}) \times 3$. By solving the updated system (Eq. 3.10) iteratively, the points are contracted while maintaining the specified maximum distance constraint among their one-ring neighbors (Fig. 3.7b).

3.2.4 Tip points preservation

It is noted that the contraction process would shorten the structure, and the leaf tips often shift along the curve skeleton, as shown in Figure 3.9a.

While the offsets between the original positions and the extracted skeleton points are typically minimal and do not significantly impact applications like 3D reconstruction, they may lead to inaccuracies in cases where high precision is required, such as plant structural traits analysis. The extracted skeleton is expected to represent the full length of the plant structures (e.g., leaf or stem length), so it becomes essential to ensure that tip points are accurately preserved. Inaccuracies in tip point locations could affect the interpretation of phenotypic traits. Therefore, it is worthwhile to pose a suitable solution to alleviate this shortening tendency during the contraction and try to keep the geometry of



(a) Plant one.

(b) Plant two.

Figure 3.10: Geodesic distance and identified leaf tip points for two different plants. The green spheres represent the source points for geometric distance calculation and the dark purple spheres indicate the detected leaf tip points. Red points are close to the source.

tip points as much as possible. To address this issue, we propose a refinement method designed to preserve the positions of the tip points during the skeleton extraction process.

For a given plant point cloud, we first identify the point in the point cloud with the lowest height value, which serves as the source point. We then calculate the geodesic distances using the heat method [Crane et al., 2017], beginning from this source point and extending to all other points in the plant point cloud. Subsequently, we perform a k-nearest neighbors search to identify the tip points based on these computed geodesic distances. Since the tip points will have the largest geodesic distances relative to their neighbors (see Algorithm 3.1), this method allows for effective identification of tip points (as illustrated in Figure 3.10).

After identifying the tip points, we impose a soft constraint by assigning a fixed value of $\mathbf{W}_L/\mathbf{W}_H$ to all detected tip points before the contraction iteration begins (as illustrated in Figure 3.1). This fixed ratio is empirically set to 1/9. The reason why we do not directly treat these points as well-contracted by setting $\mathbf{W}_L$ to 0 and $\mathbf{W}_H$ to 3.0 is that the detected tip points are not guaranteed to lie on the center line of the leaf, due to noise present in the original point cloud. Therefore, the tip points should not only be influenced by the Distance Constraint module but should also remain movable in the direction of their mean curvature normals but under constraint. By the value of $\mathbf{W}_L/\mathbf{W}_H$ fixed to 1/9, we aim to keep their original positions under a soft constraint. This approach helps balance the preservation of tip points while maintaining the overall quality of the shape of the skeleton points. This helps preserve their positions during the skeleton extraction process (Fig. 3.9b).

Algorithm 3.1: IdentifyTipPoints(**D**, **P**, k)

```
Input: The geodesic distance from source points to all other points $\mathbf{D} = \{d_1, d_2, \cdots, d_n\}$; the
       point set $\mathbf{P} = \{p_1, p_2, \cdots, p_n\}$ of original point cloud; and $k$ for $k$-nearest neighbors
       search
Output: $\mathbf{P}_{tip}$: a set of detected leaf tip points in $\mathbf{P}$

$\mathbf{P}_{tip} \leftarrow \emptyset$;
for $i \leftarrow 1$ to $|P|$ do
    $p_i \leftarrow \mathbf{P}[i]$;
    $d_i \leftarrow \mathbf{D}(p_i)$;
    $knn_i \leftarrow \mathrm{KNN}(p_i, \mathbf{P}, k)$;
    if $d_i > d_j \; \forall p_j \in knn_i$ then
        $\mathbf{P}_{tip} \leftarrow \mathbf{P}_{tip} \cup \{p_i\}$;
```

3.3 Skeleton Graph Generation

The output of Section 3.2 is a set of discrete skeleton points without any connectivity information, which can only approximate a curve skeleton. To extract phenotypic traits or segment plant organs from plant skeleton, the importance of connectivity between those extracted skeleton points is self-evident. Therefore, in this Section, we aim to build connectivity, thereby generating a curve skeleton graph that could well present the geometry structure of the input plant point cloud.

Since the Laplacian-based skeletonization technique does not reduce the number of points during contraction, the output from Section 3.2 consists of discrete skeleton points that retain the same quantity as the original input. However, continuing to process this large set of points is computationally heavy and unworthy of subsequent operations. Thus, we first downsample the extracted skeleton points using the farthest point sampling strategy [Eldar et al., 1997], lowering the computational complexity for the following steps. Then, to improve the quality of the geometry structure of the generated curve skeleton, we perform a calibration step based on the LOP operator to ensure the skeleton points are both centered and within the original shape of point cloud in Section 3.3.1. After this calibration operation, we utilize the Kruskal's MST algorithm to generate a graph that captures the connectivity between the centralized skeleton points in Section 3.3.2.

3.3.1 LOP-based calibration

The extracted discrete skeleton points based on the Laplacian-based skeletonization technique do not guarantee the centeredness and could even go outside the shape of original point cloud, especially at the regions that have relevant large differences in curvature (e.g., plant leaves, see as in Figure 3.11a), as those geometry features would be smoothed out in some degree during the iterations. We are inspired by the work of Huang et al. [2013] and Li and Wang [2018], which successfully utilized the LOP operator [Lipman et al., 2007] in curve skeleton extraction for point clouds and closed surfaces while maintaining their centeredness. Therefore, we apply the concept of the LOP operator partially and try to move each skeleton point to the L_1 median center of its corresponding region in the original point cloud to optimize the location of the curve skeleton.

The original LOP operator moves a set of points to its L_1 median position on the local region of the target object under the control of repulsion force among its neighbors [Lipman et al., 2007]. With the regularization of repulsion force, those points could have a uniform distribution on the target object while also on their optimal L_1 median position. However, in our scenario, our extracted skeleton points already have a uniform distribution to represent the structure and topology of the input point cloud. Therefore, the regularization term in the original LOP operator is unnecessary in our framework, and adding such a regularization term among the skeleton points could even cause uncurve shapes or zigzag artifacts. We modify the original LOP operator by removing its term

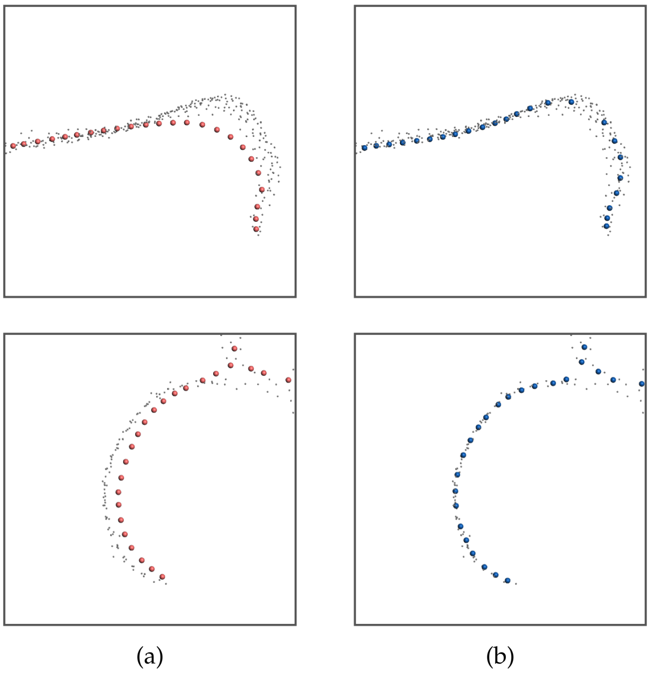


Figure 3.11: Offsets of skeleton points in regions with high curvature variations and the performance of LOP-based Calibration. (a) Offsets. (b) Maintained by LOP-based Calibration

related to repulsion force and only utilize the L_1 median term. And we compute the new position for each skeleton point x_i:

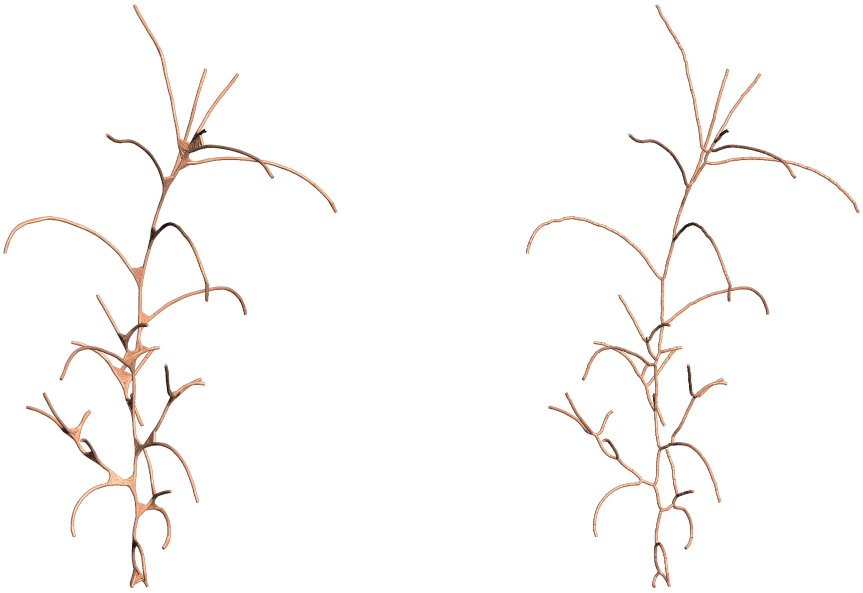
$$x_i^{New} = \frac{\sum_{j \in K} p_j \alpha_{ij}}{\sum_{j \in K} \alpha_{ij}}, \quad \text{where } \alpha_{ij} = \frac{\theta(\|p_i^t - q_j\|)}{\|x_i - q_j\|} \tag{3.11}$$

Here, p_j represents one of the k nearest neighbors of x_i in the original point cloud, and $\theta(x)$ is a smooth weight function, which is the same as the $\theta(x)$ in Equation 3.5.

3.3.2 Connectivity building

Establishing a topological relationship between discrete skeleton points is crucial for accurately reconstructing the underlying plant structure. Based on the characteristics of plant architecture, closer points are more likely to share similar semantic information and belong to the same branch in the final curve skeleton graph. This can be achieved by connecting neighboring vertices. However, simply connecting neighboring skeleton points typically produces a graph containing many duplicate edges and cycles, degrading the quality to represent the topological structure of the input plant (Fig. 3.12a). Therefore, many previous works have chosen to utilize the MST to represent the skeleton graph due to its tree structure [Du et al., 2019; Sun et al., 2021; Zhou et al., 2023].

We define the edge weight between two skeleton points, x_i and x_j, as the 3D Euclidean distance between them. Using the Kruskal's MST Algorithm, an MST is computed, removing redundant edges and cycles from the initial graph while minimizing the total edge length. The final curve skeleton graph obtained from this process could effectively represent the topological characteristics of the input plant point cloud (Fig. 3.12b). Additionally, it is important to note that the skeleton points extracted through our above procedure do not guarantee no isolated or abnormal skeleton points, which may induce noise branches in the computed MST. Therefore, we perform a pruning operation to address this issue. For an MST graph, any branch that only contains β nodes is regarded as a noise branch, thereby removing (in this study, we set β to 0.001 × num(graph vertices)).



(a) Initial graph

(b) Computed MST

Figure 3.12: Removing redundant edges and cycles by computing an MST. The brow cylinders present the skeleton.

3.4 Stem-Leaf Segmentation

With the generated curve skeleton graph of the input plant point cloud, the next step is to establish plant stem-leaf segmentation based on it. As the skeleton graph encodes the local and global geometry features of the plant, we could transform the plant stem-leaf segmentation question into an undirected graph partition problem. Therefore, we first conduct skeleton graph segmentation by using a graph-based approach. After assigning semantic and instance labels to those separated skeleton branches, we could complete stem-leaf segmentation using the nearest neighbor projection.

3.4.1 Skeleton segmentation

Given a skeleton graph $\mathcal{G} = (V, E)$, where V represents the set of skeleton points and E is the set of edges connecting these points, we classify V into three categories (as depicted in Figure 3.13a):

- Leaf vertices (V_{leaf}): Vertices that connect to only one other vertex ($\deg(v) = 1$);
- Branch vertices (V_{other}): Vertices that connect to exactly two other vertices ($\deg(v) = 2$);
- Junction vertices (V_{junction}): Vertices that connect to more than two other vertices ($\deg(v) > 2$).

Considering the typical structure of plants, we find that, in most cases, V_{leaf} are located at the tip regions of leaves. The only exception is the root node, v_{root}, which is defined as the vertex with the lowest height value among all vertices in V. Typically, v_{root} would be included in V_{leaf}; however, if a lateral stem exists at the bottom of the plant, v_{root} may be excluded from V_{leaf}. Additionally, V_{junction} vertices are typically found where the leaf petiole intersects with the stem or at junctions between the main and lateral stems.

Using this prior knowledge, we can partition the skeleton graph into semantically and structurally meaningful segments. First, we begin by traversing the graph starting from a vertex in V_{leaf}, excluding v_{root} if it belongs to V_{leaf}. The traversal follows an edge until a vertex in V_{junction} is encountered. A leaf skeleton segment is then defined by identifying the intermediate V_{other} vertices passed during

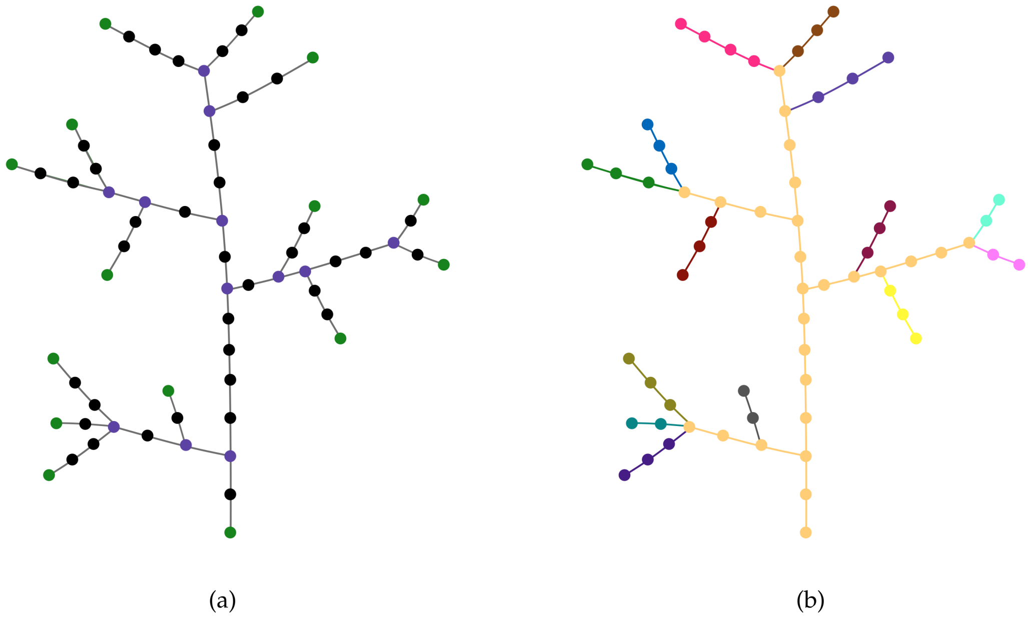


Figure 3.13: Diagram of plant skeleton segmentation. (a) Green nodes are V_{leaf}, purple nodes are V_{junction}, and black nodes are V_{other}. (b) The sketch of the plant stem and leaf skeleton segments, with each leaf marked in different colors.

this traversal, along with the starting points. The complete set of leaf segments is generated by iteratively traversing the graph from each vertex in $V_{\text{leaf}} \setminus \{v_{\text{root}}\}$ along the shortest path to a vertex in V_{junction}. The remaining portion of the skeleton graph forms the stem skeleton segment (see as in Figure 3.13b).

3.4.2 Stem-leaf detection

Having several sets of skeleton segments from skeleton graph with both semantically and instantly meanings representing plant stem skeleton and different leaves' skeletons, the detection of plant stem and leaf instances on the original point cloud can be achieved easily.

We utilize the nearest neighbor projection strategy, to assign semantic label to leaf points and stem points, and instance label to leaf points simultaneously.

Algorithm 3.2: AssignLabels(**N**, **P**)

```
Input: Skeleton graph nodes with semantic and instance labels: $\mathbf{N}$; Original point cloud
       without labels: $\mathbf{P}$
Output: Point cloud $\mathbf{P}$ with point-wise semantic and instance labels
for $i \leftarrow 0$ to $|P|$ do
    $p_i.label \leftarrow \emptyset$;
    $p_i \leftarrow \mathbf{P}[i]$;
    $closest\_node \leftarrow$ find the closest node in $\mathbf{N}$ to $p_i$;
    $p_i.label \leftarrow closest\_node.label$
```

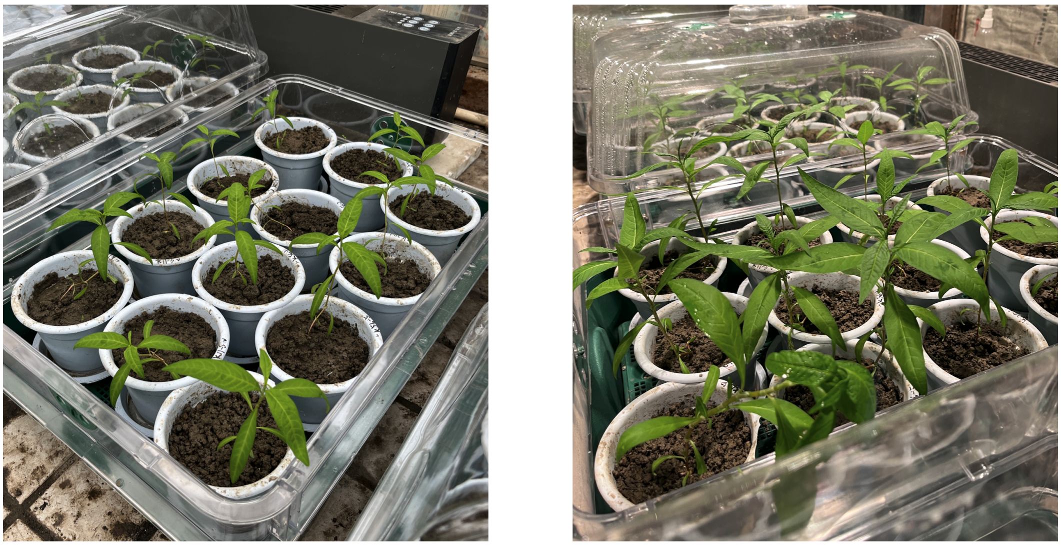


Figure 3.14: Growth environment of ***P. lapathifolium*** plants.

3.5 Photogrammetric Dataset

To evaluate the performance of our proposed framework, we reconstructed the point cloud of 56 individual ***P. lapathifolium*** seedlings, which are annual herbaceous plants widely distributed in Sichuan, China. The seeds were supplied by the College of Resources, Sichuan Agricultural University. Those ***P. lapathifolium*** were grown in individual soil pots in a greenhouse at Sichuan Agricultural University under natural diurnal temperature fluctuations and photoperiods, from December 2023 to February 2024. Figure 3.14 shows the ***P. lapathifolium*** plants in their growing environment.

3.5.1 Image data acquisition

All images were captured using commercial RGB cameras in a photo tent with uniform LED lighting, which minimized wind interference and provided consistent illumination. The camera was positioned to encircle the ***P. lapathifolium*** plants at 360° and at four different view angles: approximately 90°, 75°, 45°, 15° (see as in Figure 3.15). During the image capture, we ensured an approximate 70% overlap between adjacent images to maintain high-quality 3D point cloud reconstruction results. Approximately at least 100 images were captured for each seedling to form a comprehensive multi-view sequence.

3.5.2 Point cloud reconstruction

The 3D point clouds were automatically reconstructed from the captured multi-view image sequences using Agisoft Metashape (Version 2.1.1). The dense point cloud of each ***P. lapathifolium*** seedling, with accompanying RGB information, was stored in PLY file format. The raw data included point clouds of the pot and surrounding objects. Since segmenting individual plant point clouds and point cloud denoising are beyond the scope of this study, we manually removed the background points and significant noise. After this, we applied a Statistical Outlier Removal (SOR) filter in CloudCompare (Version 2.12.3) to further refine the point clouds. Additionally, dark points on the plant leaf edges, primarily caused by shadows, were eliminated using an RGB value filter. As the raw dense point

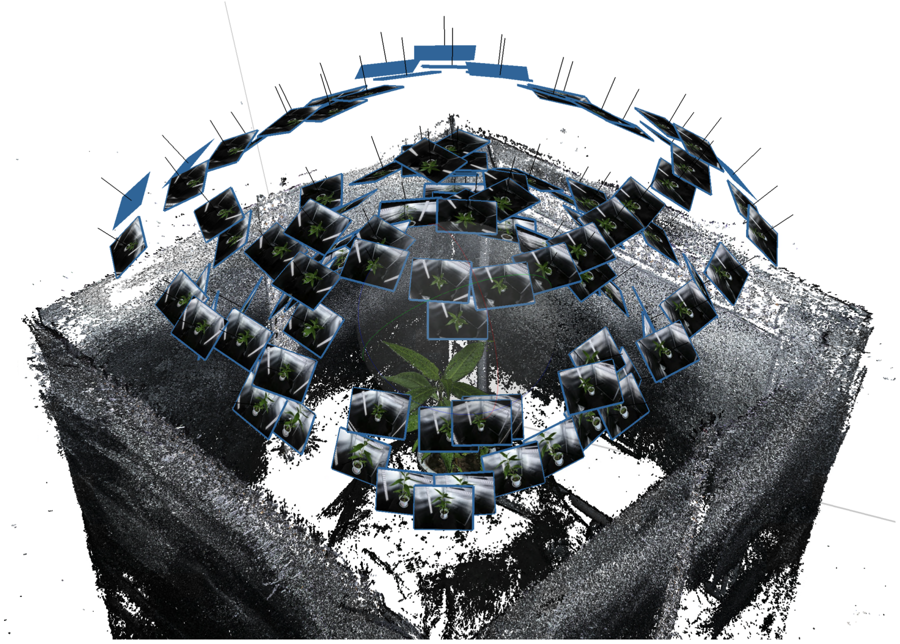


Figure 3.15: Captured multi-view image sequence for 3D reconstruction (*left*) and the reconstructed dense point cloud (*right*).

clouds were neither initially oriented in space nor consistent in scale, we first manually adjusted them to generally align vertically with the X-Y plane. Figure 3.16b shows the final point cloud stored in our dataset.

3.5.3 Point cloud annotation

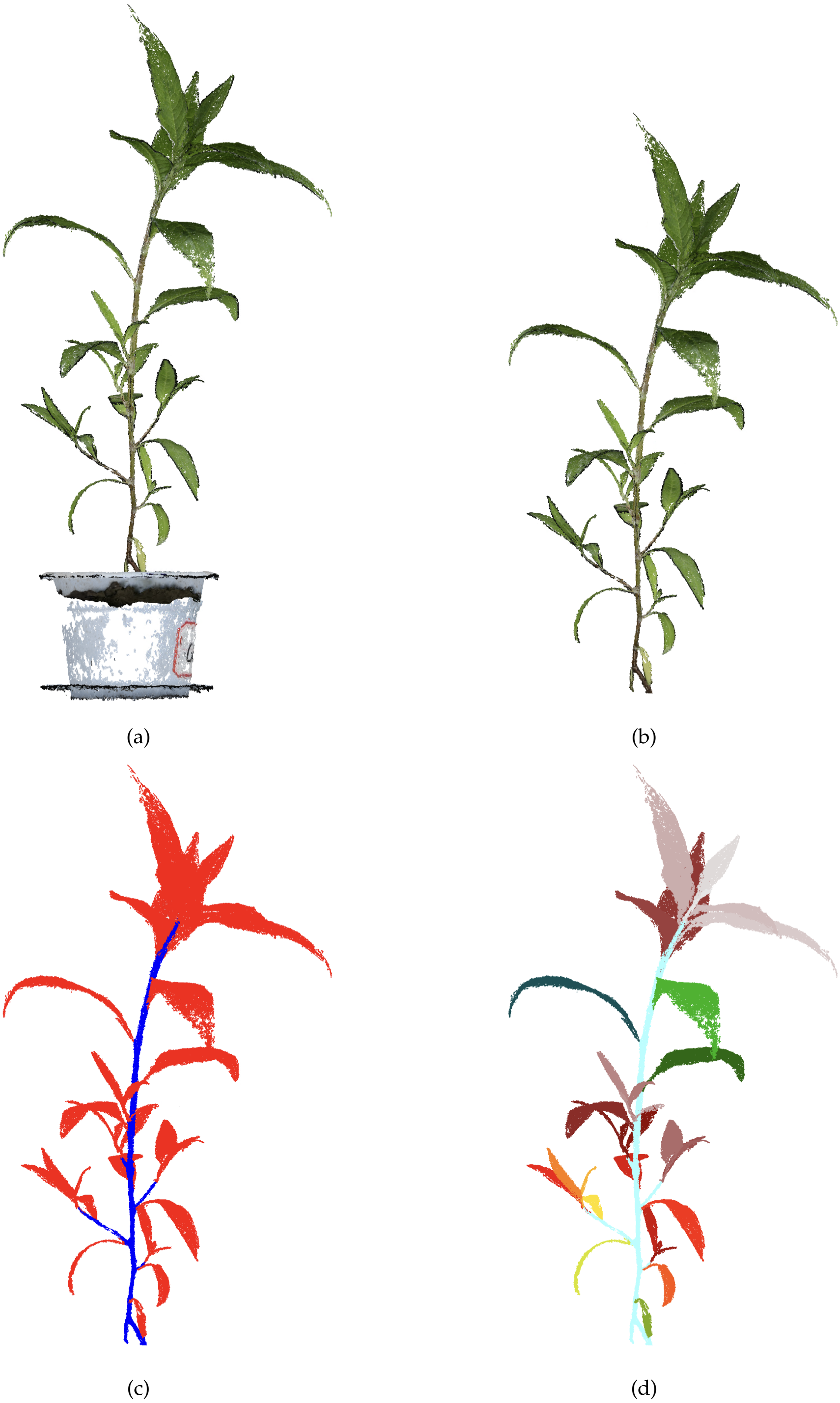
To evaluate the semantic and instance segmentation performance of our proposed framework using this dataset, we manually labeled each point in the ***P. lapathifolium*** plant point cloud as either a "stem" or "leaf" point in CloudCompare (Version 2.12.3). In addition to assigning these semantic labels, we also assigned unique instance labels to different leaves on the same plant. Figure 3.16c and 3.16d present views of one of our captured ***P. lapathifolium*** plant point clouds, illustrating the manually labeled stem and leaf points with semantic and instance labels, respectively.

3.6 Evaluation Metrics

Our proposed framework produces three outputs: the curve skeleton, plant stem-leaf semantic segmentation, and plant leaf instance segmentation. To evaluate the performance of our framework, we utilize different evaluation approaches for each of these outputs separately.

We assess the quality of the curve skeleton based mainly on visual appearance, as deriving ground-truth curve skeletons from point clouds reconstructed from real objects is challenging. Many prior works on skeleton extraction from point clouds have also evaluated their curve skeletons through visual inspection [Cao et al., 2010; Huang et al., 2013; Li et al., 2023; Wen et al., 2024]. Consequently, visual evaluation will be used to demonstrate the performance of our framework on our dataset.

In addition to visual evaluation, we employ a quantitative metric by calculating the average minimal distance from each point in the input point cloud to the curve skeleton. For the same given point cloud object, a lower average distance indicates a higher-quality curve skeleton.



(a)

(b)

(c)

(d)

Figure 3.16: Reconstructed plant point cloud and ground truth labels. (a) Plant point cloud after removing background points. (b) Final plant point cloud in our dataset. (c) Ground truth semantic labels. (d) Ground truth instance labels.

For stem-leaf semantic segmentation, we use four quantitative metrics commonly applied in point cloud semantic segmentation to evaluate performance: Precision (Prec_{sem}), Recall (Rec_{sem}), F_1-score, Intersection over Union (IoU), and overall accuracy for each semantic class. Prec_{sem} represents the proportion of correctly classified points in a semantic class relative to the total points predicted in that class. Rec_{sem} indicates the proportion of correctly classified points in a semantic class relative to the total points of that class in the ground truth. The F_1-score combines Precision and Recall using the harmonic mean to provide a composite measure, while IoU evaluates the overlap between predicted and ground truth areas for each semantic category. These metrics are computed using the following formulations:

$$\text{Prec}_{\text{sem}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.12}$$

$$\text{Rec}_{\text{sem}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.13}$$

$$F_1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.14}$$

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{3.15}$$

where True Positives (TP) denote correctly classified points, False Positives (FP) refer to points incorrectly classified as belonging to the current class, and False Negatives (FN) represent points that belong to the class but were misclassified.

For the leaf instance segmentation results, in addition to Precision and Recall, for instance, segmentation (Prec_{ins} and Rec_{ins}), we employ mean Coverage (mCov) and mean weighted Coverage (mWCov) as evaluation metrics [Ren and Zemel, 2017; Li et al., 2022a]. The IoU between each predicted instance and ground truth instance will be calculated to compute these instance metrics. A predicted instance is considered a true positive (TP) if its IoU exceeds a threshold u. In this study, we set $u = 0.5$, balancing the trade-off between looser matches being considered true positives at lower thresholds and stricter matches at higher thresholds. With the number of predicted leaf instances ($|\text{PI}|$) and the number of ground truth leaf instances ($|\text{GI}|$), Prec_{ins} and Rec_{ins} are computed as follows:

$$\text{Prec}_{\text{ins}} = \frac{|\text{TP}|}{|\text{PI}|} \tag{3.16}$$

$$\text{Rec}_{\text{ins}} = \frac{|\text{TP}|}{|\text{GI}|} \tag{3.17}$$

mCov measures the instance-wise IoU for each ground truth leaf instance, averaged across the entire point cloud. mWCov further weights the score by the number of points in each ground truth leaf instance, giving larger objects greater weight.

$$\text{mCov} = \frac{1}{|\text{GI}|} \sum_{i=|\text{GI}|}^{I} \max_j \left[\text{IoU}[\text{GI}_i, \text{PI}_j] \right] \tag{3.18}$$

$$\text{mWCov} = \sum_{i=1}^{|\text{GI}|} w_i \max_j \left[\text{IoU}[\text{GI}_i, \text{PI}_j] \right] \tag{3.19}$$

where GI_i denotes the ground truth point set for the i_{th} leaf instance, and PI_j refers to the predicted point set for the j_{th} leaf instance.

4 Results and Discussion

4.1 Results

In this section, we present the results of our framework applied to our real-captured leafy plant dataset, evaluating the quality of the extracted curve skeleton and stem-leaf segmentation. We begin by introducing the implementation details of our framework, including the development environment, required libraries, and software, and highlighting key parameters that are recommended for adjustment depending on the target plant species, along with the specific settings used in this study, as described in Section 4.1.1. Next, in Section 4.1.2, we qualitatively evaluate the performance of our plant skeleton extraction algorithm on six leafy plant point clouds from our dataset, each exhibiting different shapes. Finally, we apply our framework to the same six plant point clouds to obtain stem-leaf segmentation results, which we evaluate using relevant quantitative metrics, as detailed in Section 4.1.3.

4.1.1 Implementation details

We implemented our plant skeletonization and stem-leaf segmentation algorithm in C++ because it is well-known for its efficiency in executing computationally intensive tasks. C++ is an ideal choice for handling numerical computations, especially in the field of point cloud processing. Additionally, many of the libraries necessary for our framework are available in C++; they are listed as follows:

- **Geometry-Central** [Sharp et al., 2019a] provides a robust set of data structures and algorithms for geometric data processing. In this study, we use it to build the foundational pipeline for constructing the Tufted Laplacian operator, compute geometric distances using the heat method on point clouds, and calculate the dual vertex area for each point.
- **Eigen** [Guennebaud et al., 2010] is a widely-used C++ template library for linear algebra, and is employed in this work for matrix manipulation, sparse matrix decomposition, and solving minimum energy problems.
- **Boost Graph Library** [Siek et al., 2001] is a powerful C++ library that provides a generic interface for working with graphs. In this study uses it to build the initial graph, compute Kruskal's MST, and perform graph pruning and skeleton graph partitioning.

In addition to the above three core C++ libraries that are used directly in our program, we also adapted several useful functions from other libraries to fit our data structure. The downsampling algorithms, including farthest-point sampling and uniform sampling, are adapted from Open3D [Zhou et al., 2018] and Easy3D [Nan, 2021], respectively. Moreover, we utilize Polyscope [Sharp et al., 2019b] and Mapple [Nan, 2021] for visualization. The entire implementation was developed in the CLion 2024 environment.

Several parameters are used in our framework. In addition to the parameters we have introduced in Chapter 3, we list the remaining parameter sets in Table 4.1, which are crucial for controlling the overall quality of the extracted curve skeleton, and are recommend to be adjusted according to different plant species. Here, d_{bb} represents the maximum diagonal length of the bounding box of the input point cloud. These parameters are applied to all experiments in Chapter 3 and Section 4.1.

In all the experiments conducted in Chapter 3 and Section 4.1, the raw point clouds were first uniformly downsampled to 10,240 points to form the input point cloud. During the farthest-point

Table 4.1: The default settings of parameters

Parameter	Value	Description
k^0	8	Initial k value
Δk	4	Controlling the increment of k in each iteration
k_{max}	32	The upper limit to prevent large k
h_σ	$0.015 \cdot d_{bb}$	Sphere searching radius for weighted covariance matrix computing
d_{max}	$0.005 \cdot d_{bb}$	Maximum allowable distance between points
h_{LOP}	$0.045 \cdot d_{bb}$	Sphere searching radius for LOP operator

sampling step before LOP-based calibration, we further downsampled the contracted skeleton points to the 1% of the input point cloud (1,024).

It is worth noting that although utilizing k-nearest neighbor searching to construct the Constraint Laplacian operator allows for explicit control over neighborhood size for local triangles building, this approach can be significantly affected by the density variability of the input data. Therefore, when confronting different plant species or point cloud quality, determining an appropriate value for k can be challenging; thereby, to balance overall performance with parameter selection, we also discuss the ball query approach in Section 4.2.1, where the selection of these parameter values is also further elaborated.

4.1.2 Skeleton extraction results

To verify the effectiveness and capability of our plant skeletonization algorithm, we contract skeleton points and extract curve skeletons from six plant point clouds in our dataset, which exhibit a variety of shapes and growth directions. Among these plants, some have relatively compact architectures, the different parts of the plant are close to each other (Fig. 4.1a to 4.1c), while others have a more complex and expansive structure (Fig. 4.1d and 4.1f). Besides those plants that show complex architectures with lots of leaves and multiple lateral stems, we also include a plant that exhibits a simple architecture, characterized by fewer than 20 leaves, no lateral stem, and almost straight stems (Fig. 4.1e).

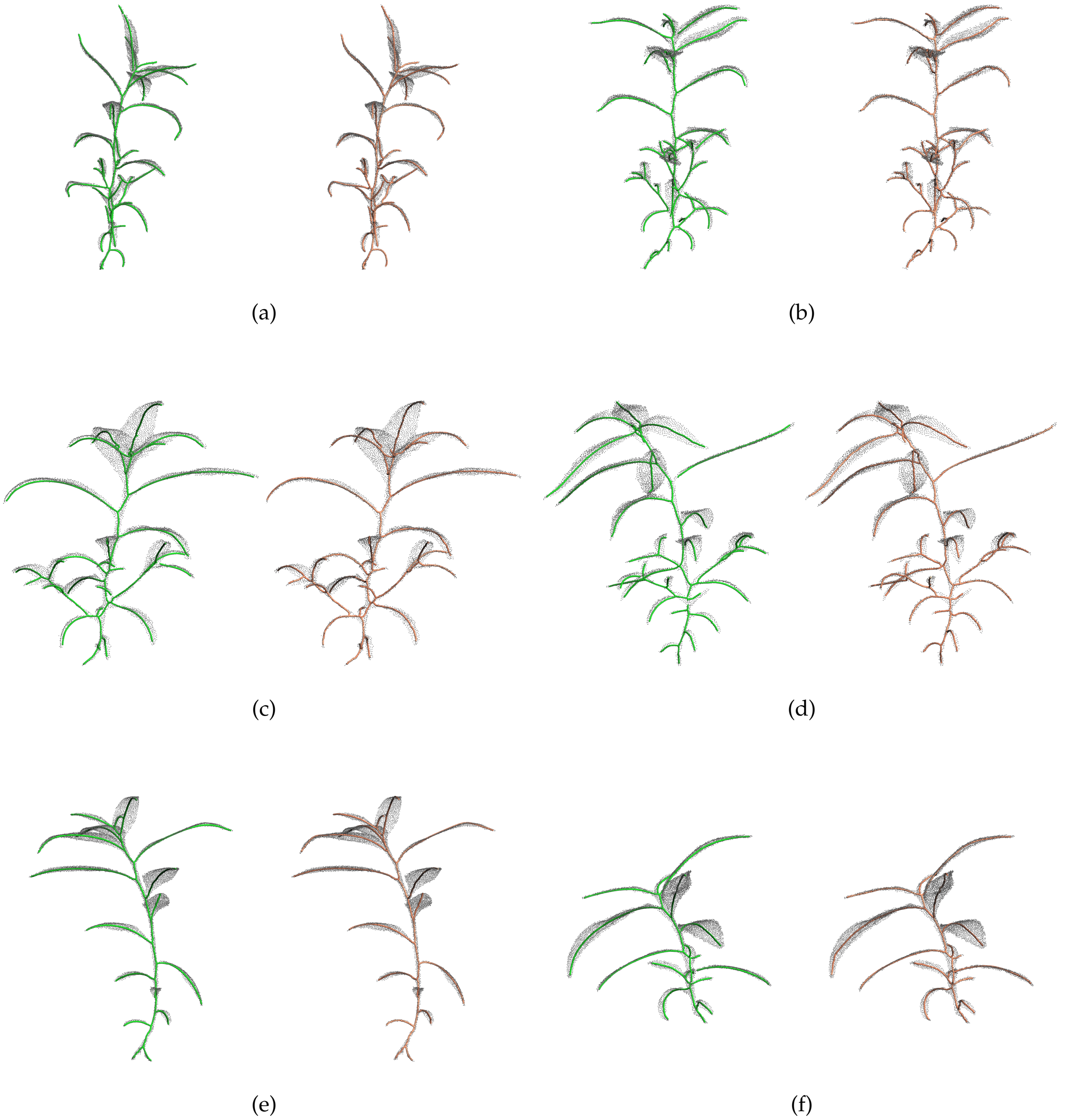
In all the above example cases, the curve skeletons are successfully extracted, demonstrating that our plant skeletonization algorithm can efficiently process a wide range of plant shapes, as illustrated in Figure 4.1. These results show that our plant skeletonization algorithm is capable of extracting curve skeletons that effectively represent the original topological structure of the plants.

However, it is observed that when the point cloud contains regions with low point density or holes due to the quality of the original data, gaps may appear in these areas in the extracted discrete skeleton points. This occurs because the point-shrinking process, being a geometry-based operation, cannot compensate for deficiencies in the original point cloud. However, these gap issues are efficiently resolved and correctly connected in the skeleton, as the discrete skeleton points retain both local and global features from other parts; therefore, during our **Skeleton Graph Generation** step, the missing connectivity information is accurately retrieved by computing the MST on the initial graph constructed by the discrete skeleton points. Overall, our plant skeletonization algorithm is capable of handling small areas with low point density or tiny holes.

4.1.3 Stem-leaf segmentation results

Semantic segmentation

Figure 4.2 presents the qualitative evaluation of the semantic segmentation results for six selected plants by our framework. According to Figure 4.2, our framework performs effective semantic seg-



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.1: Extracted plant skeletons of six selected plants. Left: Skeleton points; Right: Skeleton. Gray points represent the original plant point cloud. Green points indicate the discrete contracted skeleton points. The brow cylinders present the skeleton.

Table 4.2: The quantitative metrics of stem-leaf semantic segmentation results

Metric	Stem	Leaf	Mean
Prec_{sem} (%)	68.83	99.77	84.30
Rec_{sem} (%)	97.40	96.25	96.83
F_1-score (%)	80.66	97.98	89.32
IoU (%)	67.59	96.04	81.81

Table 4.3: The quantitative metrics of leaf instance segmentation results

	Prec_{ins} (%)	Rec_{ins} (%)	mCov (%)	mWCov (%)
Leaf	96.13	93.13	85.00	94.66

mentation, distinguishing plant stems and leaves successfully. The false segmentations are mainly observed primarily around the boundary between these two semantic classes.

Table 4.2 present the quantitative semantic results for those plants across plant stem and leaf. We can find that the Prec_{sem} and IoU show a significant disparity between the plant stem and leaf, with a lower precision for stems compared to leaves. This disparity is likely caused by the significant difference in the number of stem and leaf points, as the leaf points consistently make up the majority in all six plants. Despite this, the Rec_{sem} for both classes exceeds 95%, indicating that, based on our extracted curve skeleton, most of the stem and leaf points are correctly detected.

In our proposed framework, stem-leaf segmentation is performed by dividing the skeleton graph into specific nodes; therefore, the errors in identifying these nodes and their positions are the main reason for those false segmentations. Based on the semantic results, the misclassifications can be classified into two categories: 1) incorrect root node identification and 2) slight offsets of nodes at intersection points.

Among these results, all misclassifications at the base of the plant were due to incorrect root node identification. In our framework, the vertex with the lowest height value in the skeleton graph is simply identified as the root node. This approach is straightforward and effective in most cases. However, it fails when processing plants with downward-oriented leaves, as it can lead to the misidentification of the leaf tip vertex as the root node. This misidentification causes errors in the subsequent graph partitioning step, where the leaf skeleton segment is incorrectly classified as a stem skeleton segment, and the stem skeleton segment from the actual root node to its nearest junction is misclassified as a leaf skeleton segment. This issue is illustrated in Figure 4.2a and 4.2e.

For the remaining errors shown in Figure 4.2, the primary cause is the slight shifting of nodes at junction parts between stem and leaves. When performing point shrinking, the points in these areas are influenced by contraction forces generated by neighboring points on the same branch, as well as by points from neighboring branches. Even though we have implemented **Dynamic Neighbor Searching** and **Edge Collapse** modules to mitigate these effects, slight offsets between actual nodes still occur at these junctions, resulting in segmentation errors.

Instance segmentation

Figure 4.3 displays the qualitative evaluation of instance segmentation results for the six selected plants by using our framework, while Table 4.3 provides the corresponding quantitative metrics for leaf instance segmentation.

As illustrated in Figure 4.3, most leaf instances are accurately identified. Since both the semantic and instance segmentation results are based on the skeleton segments, their performance is quite similar, and the areas with misclassifications are largely the same. From Table 4.3, all quantitative evaluation

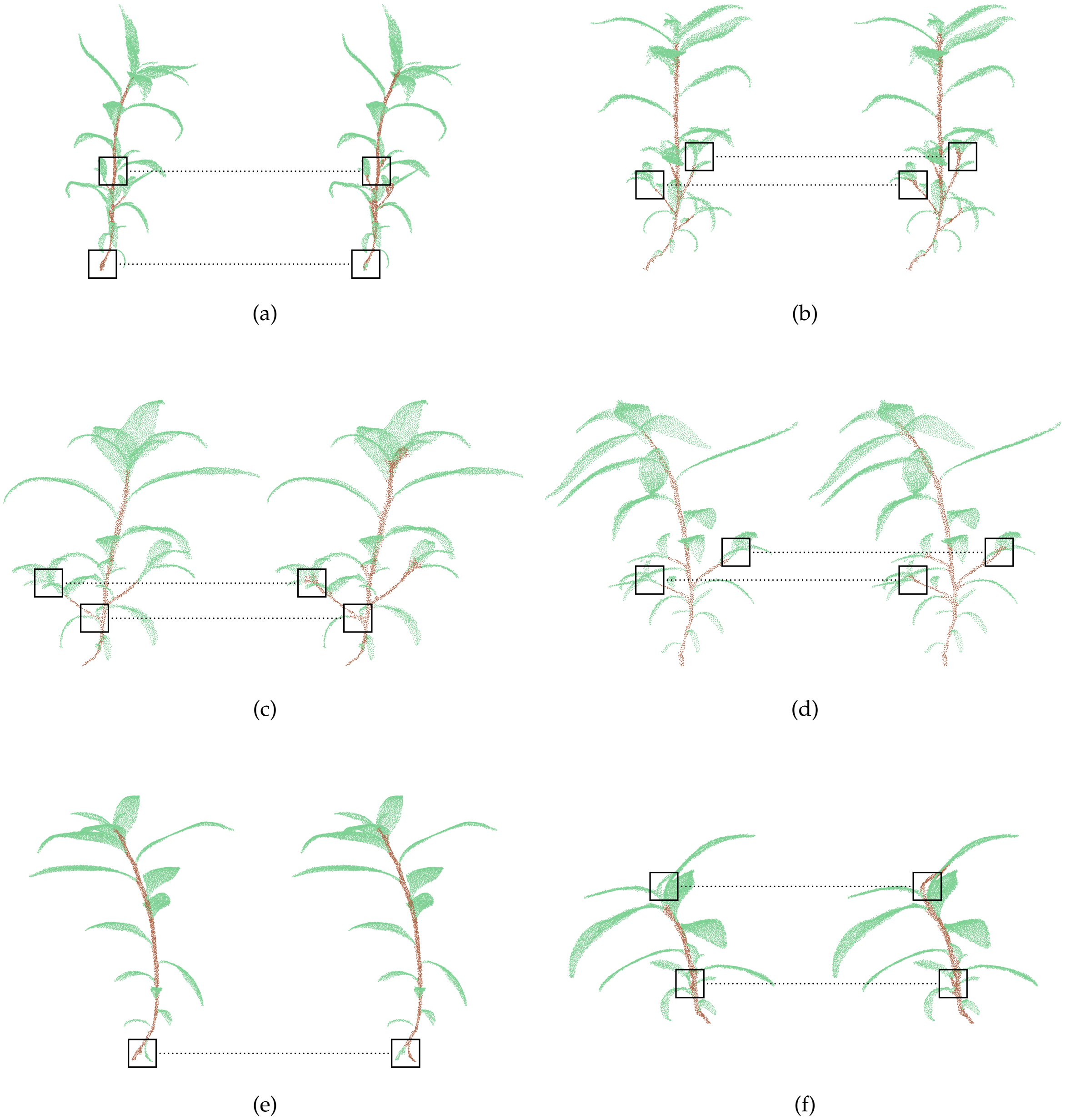
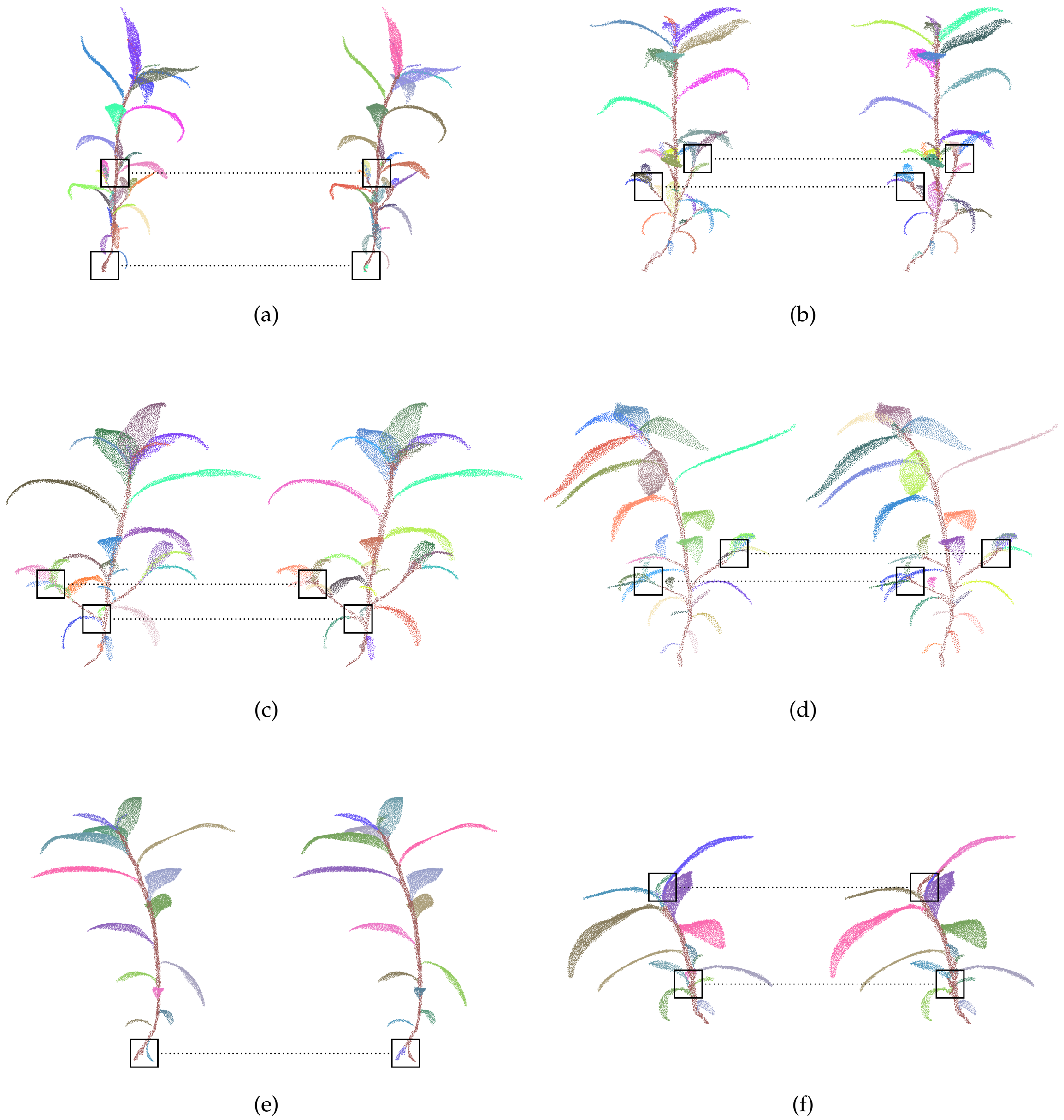


Figure 4.2: Stem-leaf semantic segmentation results of the six selected plants. Left: Ground truth; Right: Predicted results. Different point colors indicate different classes. This color scheme is consistent across the following figures.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.3: Leaf instance segmentation results of six selected plants. Left: Ground truth; Right: Predicted results

metrics exceed or equal 85%, indicating that the leaf instance segmentation, based on our extracted curve skeleton, performs effectively.

4.2 Discussion

In this section, we first examine the effects of parameter selection on the extraction of plant skeletons, as detailed in Table 4.1, and elucidate the underlying mechanisms in Section 4.2.1. Subsequently, in Section 4.2.3, we perform a comparative evaluation of the skeletons generated by our plant skeletonization algorithm against those produced by four state-of-the-art methods. This comparison underscores the superior performance and suitability of our algorithm for leafy plants, particularly ***P. lapathifolium***. Additionally, we assess the performance of our skeleton extraction algorithm across various plant species, including multiple leafy plants and trees, as detailed in Section 4.2.4. Finally, we discuss the limitations of our framework in Section 4.2.5 and explore its potential applications in Section 4.2.6.

4.2.1 Impact of parameters

There are several parameters involved in our framework. This section discusses how different parameters listed in Table 4.1 influence the results of the extracted curve skeleton.

The parameter $d_{\max}$ is related to the density of the input point cloud. Therefore, the sparse point cloud should take higher $d_{\max}$; otherwise, during the point shrinking, the points will be pulled close to each other under the forces from **Distance Constraint**, thereby losing necessary geometry features. The parameter h_{LOP} is associated with the distribution of the input point cloud. If the input point cloud has a non-uniform distribution, the value of h_{LOP} should be relatively larger to prevent the modified LOP operator from displacing points toward the center line, which can reintroduce a zigzag structure. Additionally, if the input point cloud contains fine structures, the value of h_{LOP} should be adjusted accordingly to avoid including points from other branches when computing the L_1 medial points. Therefore, according to the characteristics of our input point clouds, we empirically set $d_{\max}$ and h_{LOP} to $0.005 \cdot d_{bb}$ and $0.045 \cdot d_{bb}$, respectively.

For the remaining four parameters (h_σ, k^0, Δk, and $k_{\max}$), we discuss them in detail in the following sections. Additionally, rather than explicitly fine-tuning the parameter k, we propose an alternative approach for managing **Dynamic Neighbor Searching** by employing a ball query strategy, as discussed in Section 4.2.1. This method provides greater generality and robustness when applied to diverse plant point clouds with varying densities and quantities. While the performance of the ball query approach is slightly poor on our specific dataset, it demonstrates broader applicability beyond the particular plant shapes and point cloud densities utilized in our experimental framework.

The value of h_σ

The parameter h_σ is a crucial element in the **Directionality-based Weighting** module, as it determines the search area for the smooth_σ calculation (see as in Equation 3.5). During the point-shrinking process, all points are gradually contracted into a more concise shape. Due to the varying structures within a point cloud, some points cluster together more quickly, forming local branches. A larger h_σ allows for evaluating the contraction degree over a broader area, efficiently mitigating the influence of small, abnormal local branches that may form during the iteration. This helps to avoid the misidentification of under-contracted points. However, setting h_σ too high can also be problematic. It may lead smooth_σ to be calculated across multiple branches rather than just a single branch, and well-contracted points may not be identified in time, triggering over-contraction. Additionally, a very high h_σ may also result in under-contraction; considering a larger region, the overall smooth_σ could exceed 0.9, while the central area remains under-contracted.

On the other hand, if h_σ is too small, it may falsely identify under-contracted points as well-contracted, causing abnormal branches to form or resulting in insufficient contraction results. We tested h_σ values ranging from $0.005 \cdot d_{bb}$ to $0.135 \cdot d_{bb}$, with the results presented in Figure 4.4.

From these tests, we can conclude that when h_σ is too small (e.g., $0.005 \cdot d_{bb}$), many under-contracted points are mistakenly identified as well-contracted, preventing the point cloud from contracting into a line-like structure. Conversely, when h_σ is too large, points in junction areas are not correctly identified as well-contracted, causing them to merge with nearby branches and resulting in gaps. Based on our experiments, we selected $h_\sigma = 0.015 \cdot d_{bb}$ as the optimal threshold value.

Parameters in Dynamic Neighbor Searching

In the process of plant skeletonization, the k values, controlled by k^0, Δk, $k_{\max}$ (Eq. 3.4), in **Dynamic Neighbor Searching**, play a crucial role. This value determines the constraint Laplacian operator $\mathcal{L}$ and directly affects the contraction process.

First, we will discuss how to determine $k_{\max}$. The value of $k_{\max}$ controls the maximum value of k during the contraction iterations. If $k_{\max}$ is set too low, k will be constrained within a small range, which helps preserve local geometric features during the contraction process at a fine scale. However, this can lead to the formation of abnormal branches in the final skeleton, as a low $k_{\max}$ prevents the algorithm from capturing geometric features at larger scales. As discussed in Section 3.2.2, when $k_{\max}$ is set to a relatively high value, the local geometric features are initially preserved, and as the k value dynamically increases, these abnormal branches, due to their small scale, are insufficiently be detected by the **Directionality-based Weighting** module and can be smoothed out in the subsequent iterations.

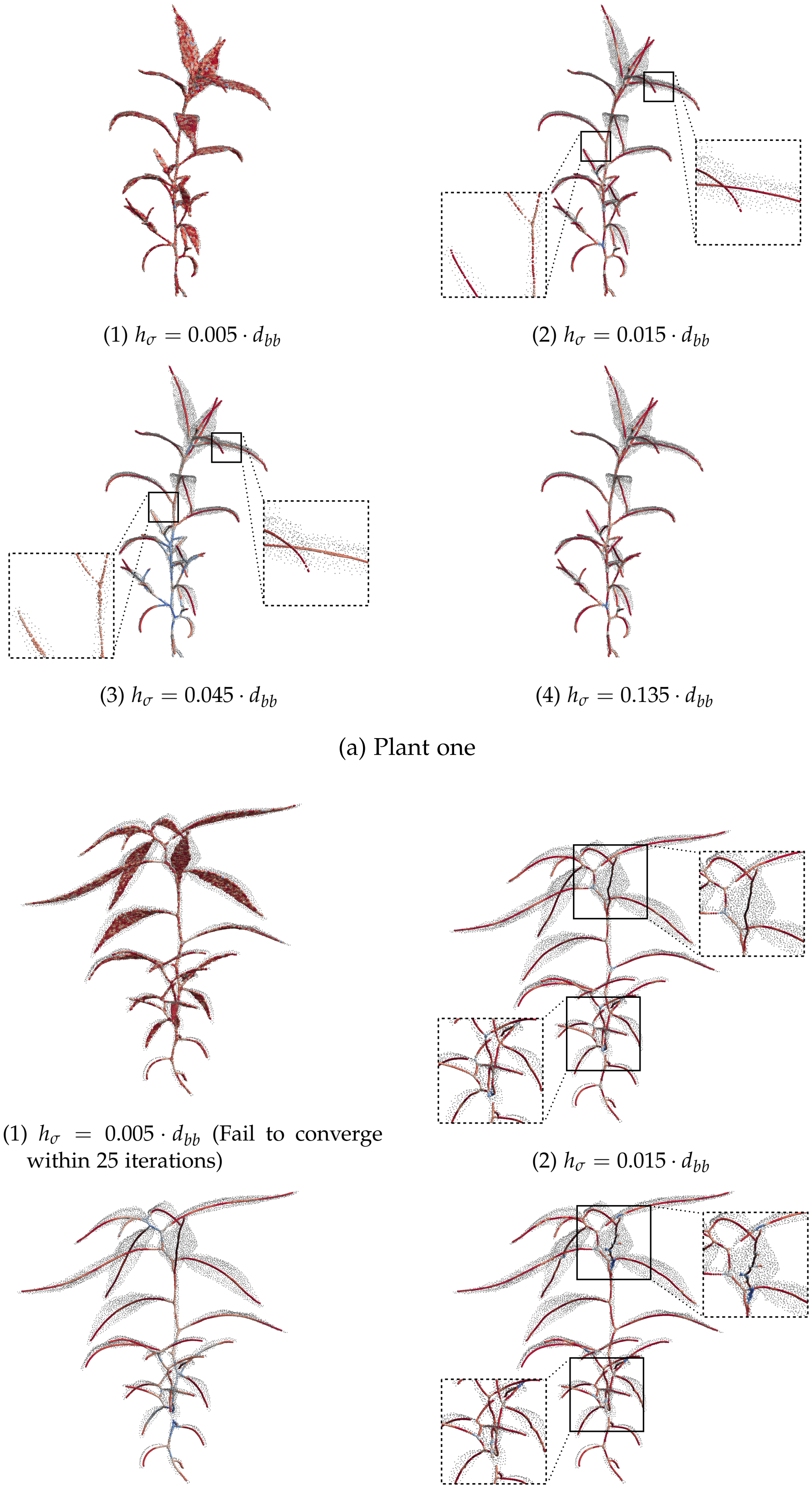
Setting a very high $k_{\max}$ does not significantly affect the Tufted Laplacian operator. As k increases beyond a certain threshold, the one-ring neighborhood of each point on the intrinsic Delaunay triangulation, formed by the union of local triangles built upon each point and its k nearest neighbors, stabilizes. As a result, the constructed $\mathcal{L}$ remains largely unchanged, which has minimal impact on the final skeletonization result.

Therefore, it is important to find a suitable $k_{\max}$ that balances the quality of the extracted skeleton, ensuring that all local and global features are enabled to be detected by the Tufted Laplacian operator and the computational cost. To achieve this, it is necessary to resolve the implicit relationship between the k value and the size of the one-ring neighborhood, which will help determine the appropriate $k_{\max}$.

To address this issue, we compute the Tufted Laplacian operator (without incorporating the **Dynamic Neighbor Searching** and **Edge Collapse** modules) on the original point cloud for various k values, ranging from 4 to 40, across 12 randomly selected plant point clouds from our dataset. We then evaluate the sparsity of the Tufted Laplacian operator, defined as the average number of non-zero elements per row (see Fig. 4.5). The results reveal that the one-ring neighborhood becomes stable when $k \geq 32$. In addition to this intuitive and straightforward experiment, prior studies have demonstrated that the minimum value of k required to construct a reliable Laplacian operator using the k-nearest neighbors method is 30 [Belkin et al., 2008]. To mitigate the computational overhead associated with larger k values, we can reasonably restrict k to this range. Based on these findings, we set $k_{\max}$ to 32 in our practice.

In addition, since most plant point clouds in our dataset meet the termination condition within eight iterations, we set Δk to 4. This setting allows the algorithm to capture local geometric features in the early stages of iteration while considering higher-level geometric structures in the later stages.

Next, we discuss the choice of the initial neighborhood size, k^0. A smaller k^0 aims to preserve detailed geometric features during contraction. Although the addition of Δk in subsequent iterations can help eliminate abnormal or noisy structures, there are cases where this fails. For instance, **Directionality-based Weighting** module might incorrectly treat abnormal branches, which are formed by preserving



(1) $h_\sigma = 0.005 \cdot d_{bb}$

(2) $h_\sigma = 0.015 \cdot d_{bb}$

(3) $h_\sigma = 0.045 \cdot d_{bb}$

(4) $h_\sigma = 0.135 \cdot d_{bb}$

(a) Plant one

(1) $h_\sigma = 0.005 \cdot d_{bb}$ (Fail to converge within 25 iterations)

(2) $h_\sigma = 0.015 \cdot d_{bb}$

(3) $h_\sigma = 0.045 \cdot d_{bb}$

(4) $h_\sigma = 0.135 \cdot d_{bb}$

(b) Plant two

Figure 4.4: Contracted skeleton points by using different h_σ values. Gray points represent the original plant point cloud. Red points indicate the higher smooth_σ value.

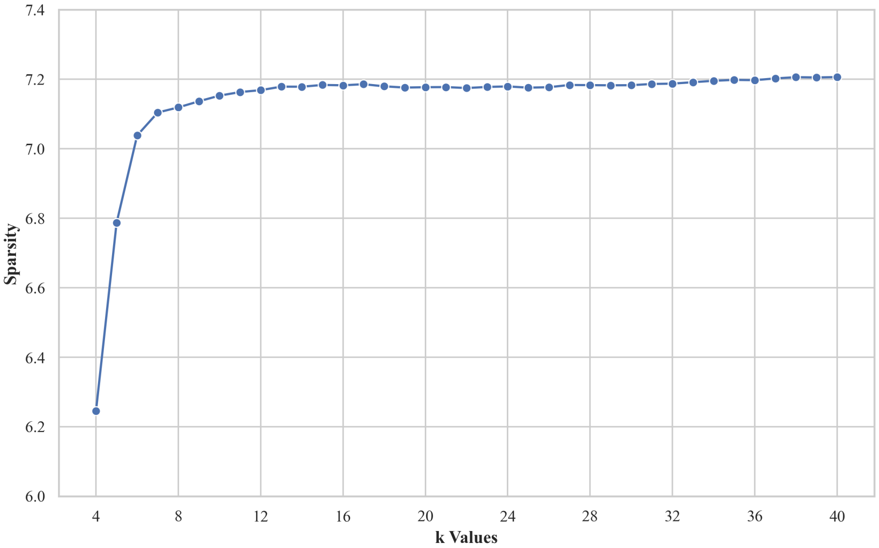


Figure 4.5: The Sparsity of the Tufted Laplacian operator.

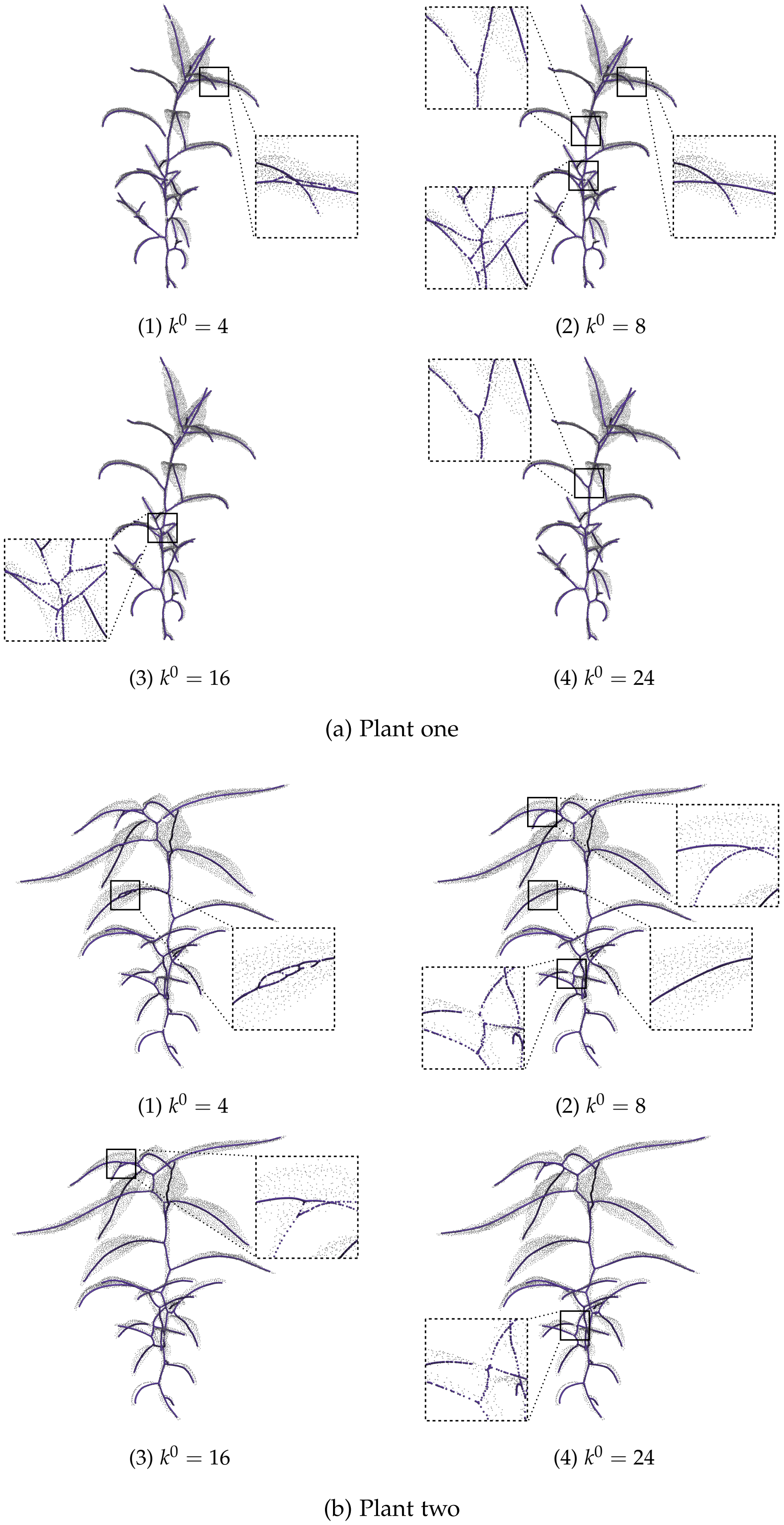
detailed geometric features at a lower scale, as well-contracted parts, introducing errors into the final skeleton.

Figure 4.6 shows the skeletonization results (contracted discrete skeleton points) for different k^0 values, with Δk and k_{max} set as previously described, and all other modules from Section 3.2 are enabled. As illustrated in Figure 4.6, when $k^0 = 4$, several noisy branches are generated. For $k^0 = 16$ and $k^0 = 24$, the larger one-ring neighborhood causes two close branches to merge, resulting in the unwanted displacement of nodes at the junction points. However, when $k^0 = 8$, these issues are effectively alleviated. Based on our experiments on different plant point clouds, we selected $k^0 = 8$ as the optimal value.

Different neighbors searching strategy

The mechanism and effectiveness of the parameters used in the default strategy of **Dynamic Neighbor Searching** have been discussed in the previous section. Although the default k-nearest neighbors strategy allows users to explicitly control the neighborhood size to construct $\mathcal{L}$, setting a low k value helps preserve fine details in the target plant point cloud. However, this strategy is highly sensitive to the characteristics of the input point cloud. The k-nearest neighbors strategy performs effectively when applied to point clouds with both a large number of points and complex fine structures. For point clouds with a large number of points but fewer fine structural details, a lower k value may introduce noisy branches, as the initial contraction occurs at a finer local level, negatively impacting the final contraction outcome (see as in Fig. 4.7). For our example plant point cloud, we can find that when the input point number is up to $51,200$, several noisy branches occur, and when the input point number is up to $102,400$, the contracted points cannot efficiently approximate the skeleton. Therefore, by using the k-nearest neighbors strategy, downsample operation is typically necessary before being processed by our framework. We recommend a downsample size of $10,240$ points based on empirical observations, which has consistently performed well under our default parameter settings.

However, for some plants with significant structural details, such downsampling operations may degrade their shape representation, leading to a substantial loss of geometric information and, consequently, impairing the final skeletons. Additionally, the k-nearest neighbors strategy is sensitive to noise points, meaning that the input point cloud must be sufficiently denoised for optimal performance. Although the advantage of explicit operation and explicitly preserving fine details, these limitations still significantly limit the generalizability of the k-nearest neighbors strategy.



(1) $k^0 = 4$

(2) $k^0 = 8$

(3) $k^0 = 16$

(4) $k^0 = 24$

(a) Plant one

(1) $k^0 = 4$

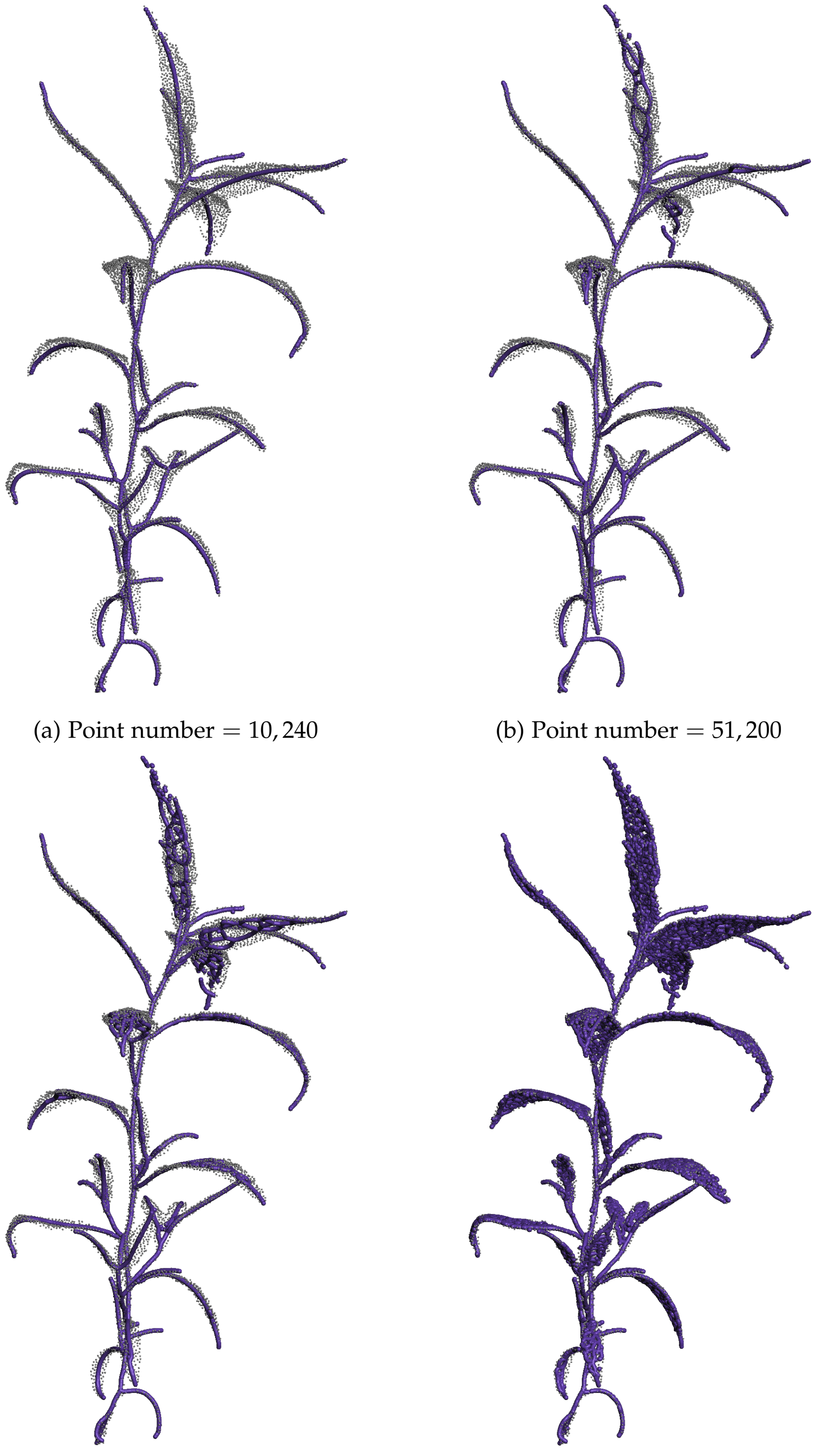
(2) $k^0 = 8$

(3) $k^0 = 16$

(4) $k^0 = 24$

(b) Plant two

Figure 4.6: Contracted skeleton points using different k^0 values. Gray points represent the original plant point cloud. Purple points indicate the discrete contracted skeleton points. This color scheme is consistent across the following figures.



(a) Point number $= 10,240$

(b) Point number $= 51,200$

(c) Point number $= 102,400$

(d) Point number $= 512,000$

Figure 4.7: Performance of *k*-nearest neighbors strategy on different input point numbers.

Table 4.4: The default settings of ball query strategy

Parameter	Value	Description
rad^0	$0.015 \cdot d_{bb}$	Initial *rad* value
Δrad	$0.005 \cdot d_{bb}$	Controlling the decrement of *rad* in each iteration
$rad_{\min}$	$0.005 \cdot d_{bb}$	The bottom limit to prevent large *rad*

The ball query strategy offers a chance to address these limitations by mitigating the impact of both input point cloud size and noise points. As the contraction iterations proceed, points tend to cluster together. Consequently, if the search radius remains fixed or increases (as suggested by Equation 3.4), the number of neighbors per vertex grows significantly. This can lead to losing fine details over time and increase computational complexity. Concerning these, we propose a new dynamic scheme for ball query strategy:

$$rad^{t+1} = \begin{cases} rad^t & rad^t - \Delta rad < rad_{\min} \\ rad^t - \Delta rad & \text{otherwise} \end{cases}, \tag{4.1}$$

where rad^{t+1} is the radius used for the ball query to build the initial union triangles for the Tufted Laplacian operator contraction at iteration $t+1$. The parameter Δrad controls the decrement of *rad* in each iteration, while $rad_{\min}$ is the lower limit to prevent excessively small values. Similar to k^0, rad^0 is recommended to be slightly increased according to the shape characteristics of different plant point clouds. Typically, for plants without relatively wide and broad shapes, a lower rad^0 is suggested (e.g., $0.015 \cdot d_{bb}$ to $0.025 \cdot d_{bb}$). On the other hand, for those with wider and broader shapes, rad^0 is usually recommended to be set within a higher range (e.g., $0.035 \cdot d_{bb}$ to $0.045 \cdot d_{bb}$).

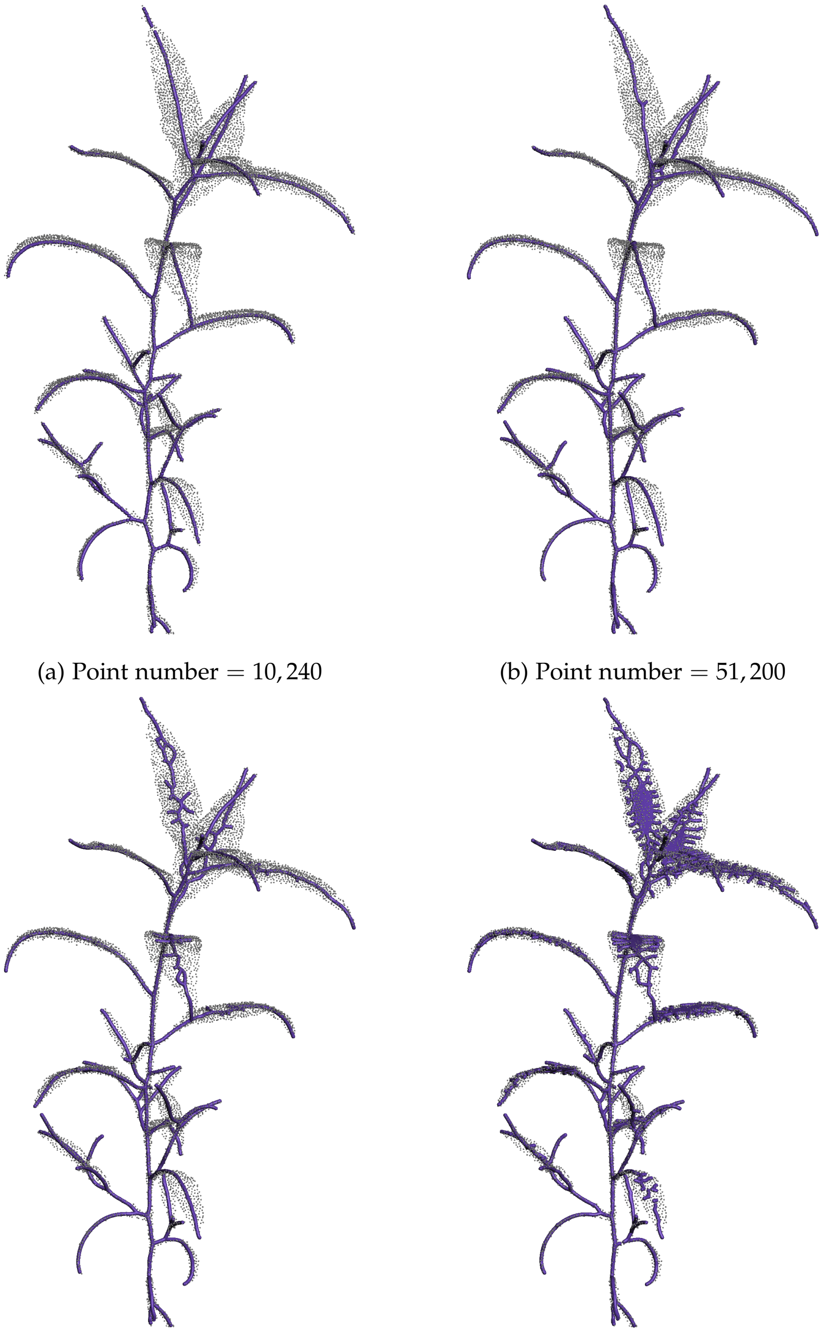
Figure 4.8 demonstrates the performance of the ball query strategy with varying input point counts. The ball query relevant parameters are empirically configured according to Table 4.4. The results indicate that even with an input size of 102,400 points, the contracted points can still accurately approximate the skeleton shape. This finding suggests that the ball query strategy can effectively mitigate some of the limitations inherent in the k-nearest neighbors strategy to a certain extent.

However, the ball query strategy is not without limitations. Using the ball query approach to control **Dynamic Neighbor Searching** can be imprecise, as the neighbors selected for each vertex vary depending on the local point cloud density. This variability can cause slight shifts at junction regions (as shown by comparing Figure 4.7a and 4.8a), where the query ball covers the entire junction, leading to positional shifts. Reducing the query ball size may avoid such shifts but introduces noisy branches, similar to the issue discussed in Section 4.2.1. Additionally, in cases of input point clouds with low density, the ball query strategy may fail to find a sufficient number of neighbors to construct local triangles effectively.

In conclusion, users should select an appropriate neighbor searching strategy based on the characteristics of the input point cloud. The k-nearest neighbors strategy is appropriate for plant point clouds containing moderate or complex fine details. Conversely, for plant point clouds that have fewer structural details (e.g., seedlings with only a single stem and a few leaves) or simply with a large size, the ball query strategy is more suitable. In general, the ball query strategy is capable of producing acceptable skeletons across most cases. However, the k-nearest neighbors strategy can further enhance skeleton quality when applied to compatible plant point clouds.

4.2.2 Computation complexity

Computation complexity is a critical factor in evaluating the performance of any program. Therefore, in this section, we qualitatively analyze the total computation time, including skeleton extraction and skeleton-based segmentation.



(a) Point number = 10,240

(b) Point number = 51,200

(c) Point number = 102,400

(d) Point number = 512,000

Figure 4.8: Performance of ball query strategy on different input point numbers. Note: The plant in Figure 4.8d fails to converge within 25 iterations.

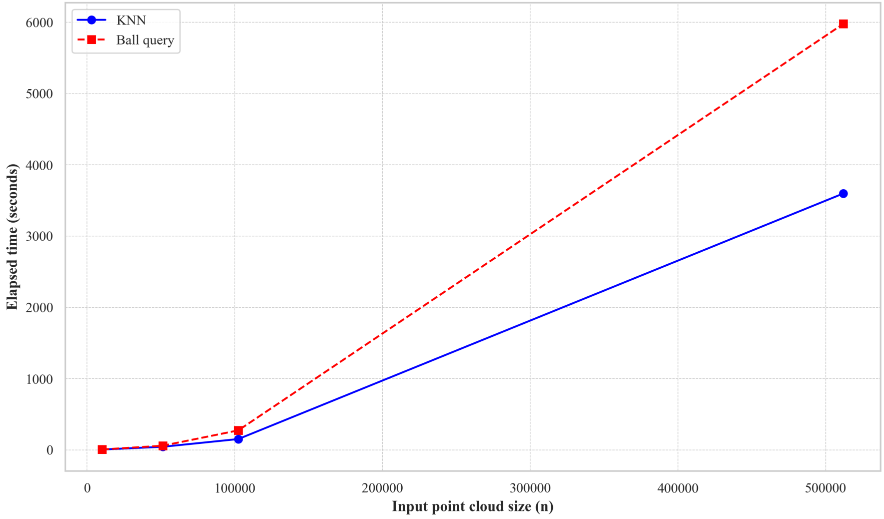


Figure 4.9: The computation time under different input point cloud size.

The core component of our framework is LBC, making the computation complexity highly sensitive to the size of the input point cloud. To assess this, we use a single plant as a reference example and evaluate the computation complexity of its downsampled versions with $10,240$, $51,200$, $102,400$, and $512,000$ points, applying both the k-nearest neighbors and ball query strategies. Please note that this experiment is conducted on a PC with an Apple M3 Max 14-core CPU.

The results are illustrated in Figure 4.9, showing the elapsed time for each configuration. Our findings reveal that the relationship between computation complexity and input point size resembles a exponential function. Based on these results, we conclude that our framework can process point clouds of approximately $100,000$ points in under 5 minutes. Typically, individual plant point clouds can be downsampled to around $100,000$ points while preserving overall geometric information, indicating that our framework can handle most cases within an acceptable time frame.

4.2.3 Skeleton extraction comparisons

Since the plant skeletonization algorithm is the core component of our framework and our main contribution, it is essential to demonstrate its advantages by comparing it with several state-of-the-art methods for extracting curve skeletons from point clouds. The methods selected for comparison include two widely recognized skeletonization techniques, LBC [Cao et al., 2010][1] and L_1-medial [Huang et al., 2013][2], both of which have been extensively used in the literature. These methods were originally designed for extracting curve skeletons from uniform 3D objects. In addition to these general-purpose methods, we also compare our algorithm with two skeletonization techniques designed initially for trees: AdTree [Du et al., 2019][3] and WoodSKE [Wu et al., 2021][4]. We obtained the codes or executable programs for these methods from the authors' repositories on GitHub or their websites and used the default parameters to extract curve skeletons from point clouds. The settings for our skeleton extraction algorithm are provided in Table 4.1, with the k-nearest neighbors strategy selected as the neighborhood searching method. For WoodSKE, the output consists of discrete skeleton points without connectivity information. Therefore, we approximate the skeleton using the discrete points directly and use this approximation to compare with other methods. Notably, we did not compare the computational time of these methods, as they were run in different environments.

[1] LBC: https://github.com/taiya/cloudcontr
[2] L_1-medial: https://shihaowu.net
[3] AdTree: https://github.com/tudelft3d/AdTree
[4] WoodSKE: https://github.com/wbx1727031/WoodSKE

Table 4.5: The quantitative metrics of skeleton extraction results

Plant Number	Ours	LBC	L_1-medial	AdTree	WoodSKE
A	0.021065	0.026008	0.032905	0.006505	0.037584
B	0.031357	0.037931	0.042576	0.007704	0.041590
C	0.031330	0.035297	0.042870	0.009241	0.049664
D	0.029281	0.031664	0.033632	0.008823	0.035311
Average	0.028258	0.032725	0.037996	0.008068	0.041037

In this comparison, all input plant point clouds are preprocessed, including normalization to a sphere with a diameter of 1.6 units. Additionally, all input point clouds are downsampled to $10,240$ points. Figure 4.10 to 4.13 present a visual comparison of our skeletonization approach against the aforementioned methods.

In addition to visual evaluation, we utilize the average minimal distance from each point in the input point cloud to the corresponding skeleton. From Table 4.5, we observe that the skeletons generated by AdTree achieve the lowest average minimal distance for all plant point clouds. However, visual inspection reveals that this result is due to the complex and dense skeleton representation on plant leaves. Therefore, while AdTree yields the lowest distance values, its skeleton representation, especially of the leaves, is inadequate. Besides AdTree, our skeletons are ranked as the second lowest value for all plant point clouds, outperforming all other three algorithms.

Based on both visual and quantitative comparisons, our skeleton extraction algorithm demonstrates superior structural integrity, location accuracy, and connectivity between the main stem, lateral stems, and leaves. Several factors contribute to this improvement:

1. **Constraint Laplacian Operator:** By incorporating the local and global geometric features hierarchically, our approach maintains shrinkage direction, which ensures the topological and geometric accuracy and the integrity of the skeleton.

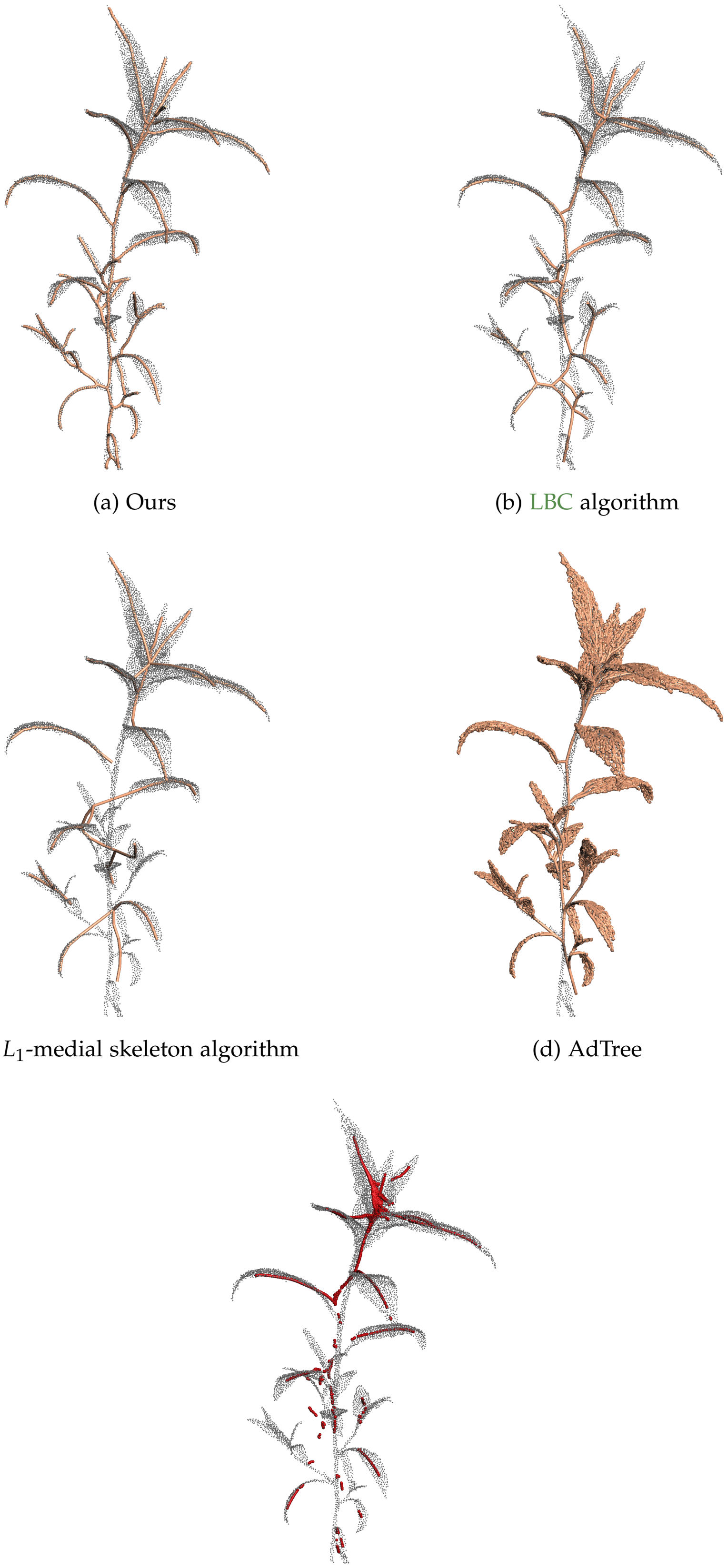
2. **Adaptive Contraction and Tip Points Preservation:** This step ensure the completeness of the line-like object during the contraction process while avoiding the creation of skeleton branches. It also maintains geometric accuracy and ensures connectivity between plant components (i.e., main stem, lateral stems, and leaves), avoiding gaps.

3. **LOP-based Calibration:** We utilize the modified LOP operator to calibrate the location accuracy of the skeleton.

4.2.4 Skeleton extraction evaluation on diverse species

The results shown in Section 4.1.2 demonstrate that the skeleton extraction algorithm in our framework can compute satisfactory skeletons for the specific plant species used in our tests (***P. lapathifolium***). To further evaluate and illustrate the generalizability of the skeleton extraction algorithm, we test its performance on several plant point cloud datasets, including four leafy plants (Maize from Pheno4D [Schunck et al., 2021], and Pepper, Ribes, and Rose from PLANesT-3D [Mertoğlu et al., 2024]), as well as four different trees from TreeML-Data [Yazdi et al., 2024].

We employ the ball query strategy for the four types of leafy plants, which exhibit significant structural variation, to manage **Dynamic Neighbor Searching** and downsample the point clouds to $51,200$ points. This downsample operation balances computation complexity and preserves the original geometric information. Additionally, to handle the small structures in the Ribes plant and the wider, broader shapes in the Rose plant, we set rad^0 to $0.035 \cdot d_{bb}$ for robustness. All other parameters are set to their default values.



(a) Ours

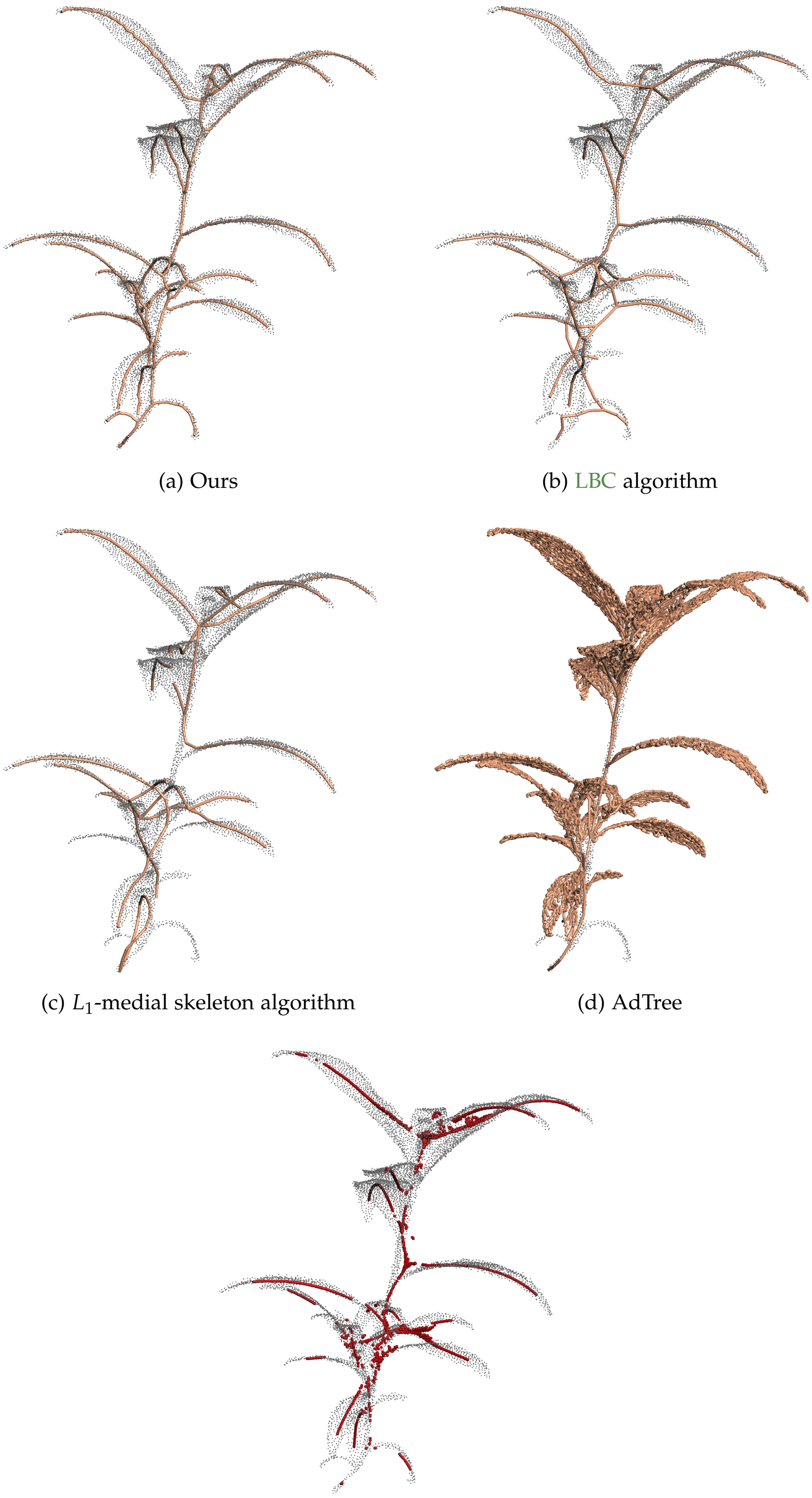
(b) LBC algorithm

(c) L_1-medial skeleton algorithm

(d) AdTree

(e) WoodSKE (Discrete contracted skeleton points)

Figure 4.10: Skeletonization results comparison (Plant A). Gray points represent the original plant point cloud. Red points indicate the discrete contracted skeleton points. The brow cylinders present the skeleton. This color scheme is consistent across the following figures.



(a) Ours

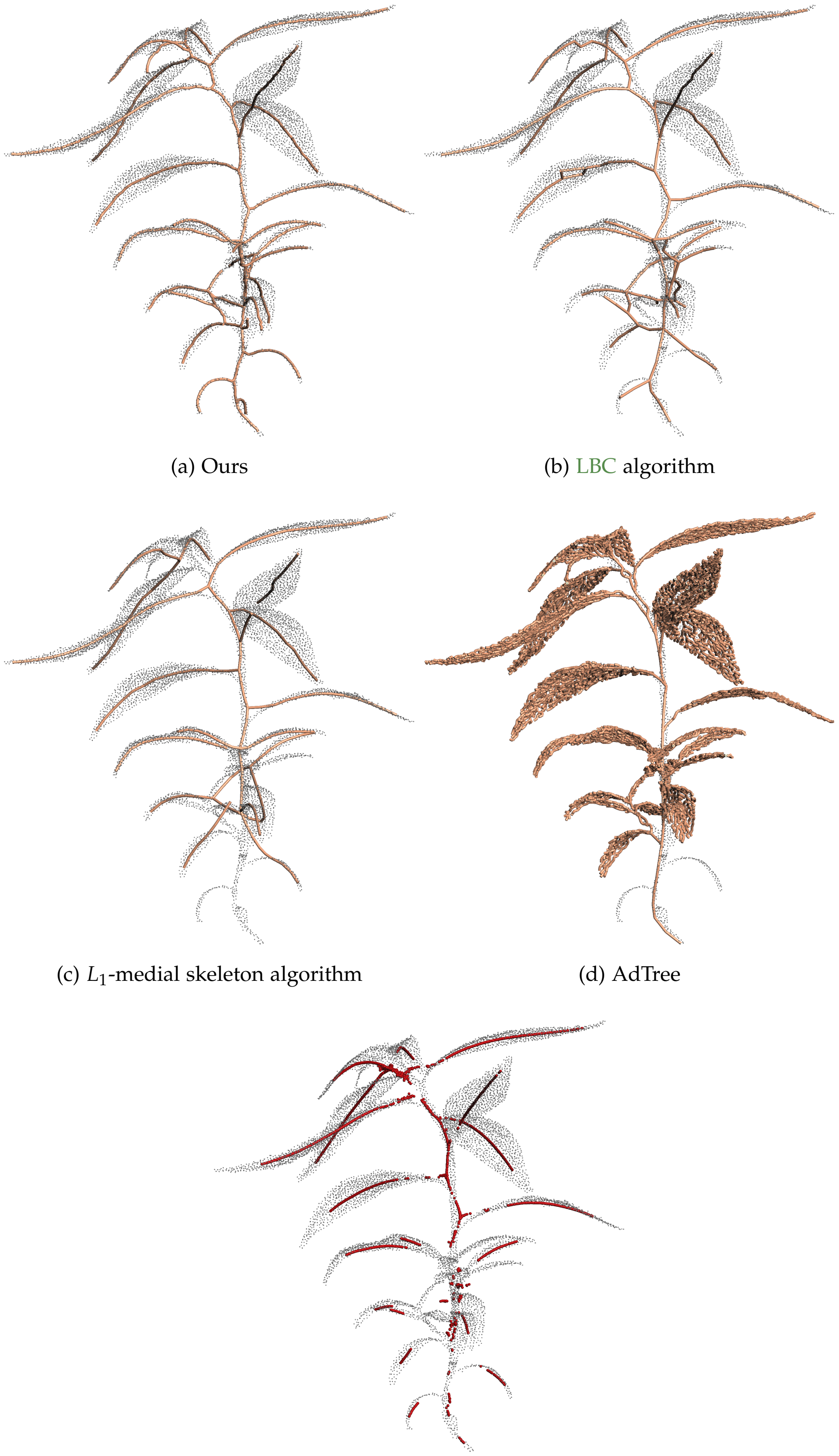
(b) LBC algorithm

(c) L_1-medial skeleton algorithm

(d) AdTree

(e) WoodSKE

Figure 4.11: Skeletonization results comparison (Plant B).



(a) Ours

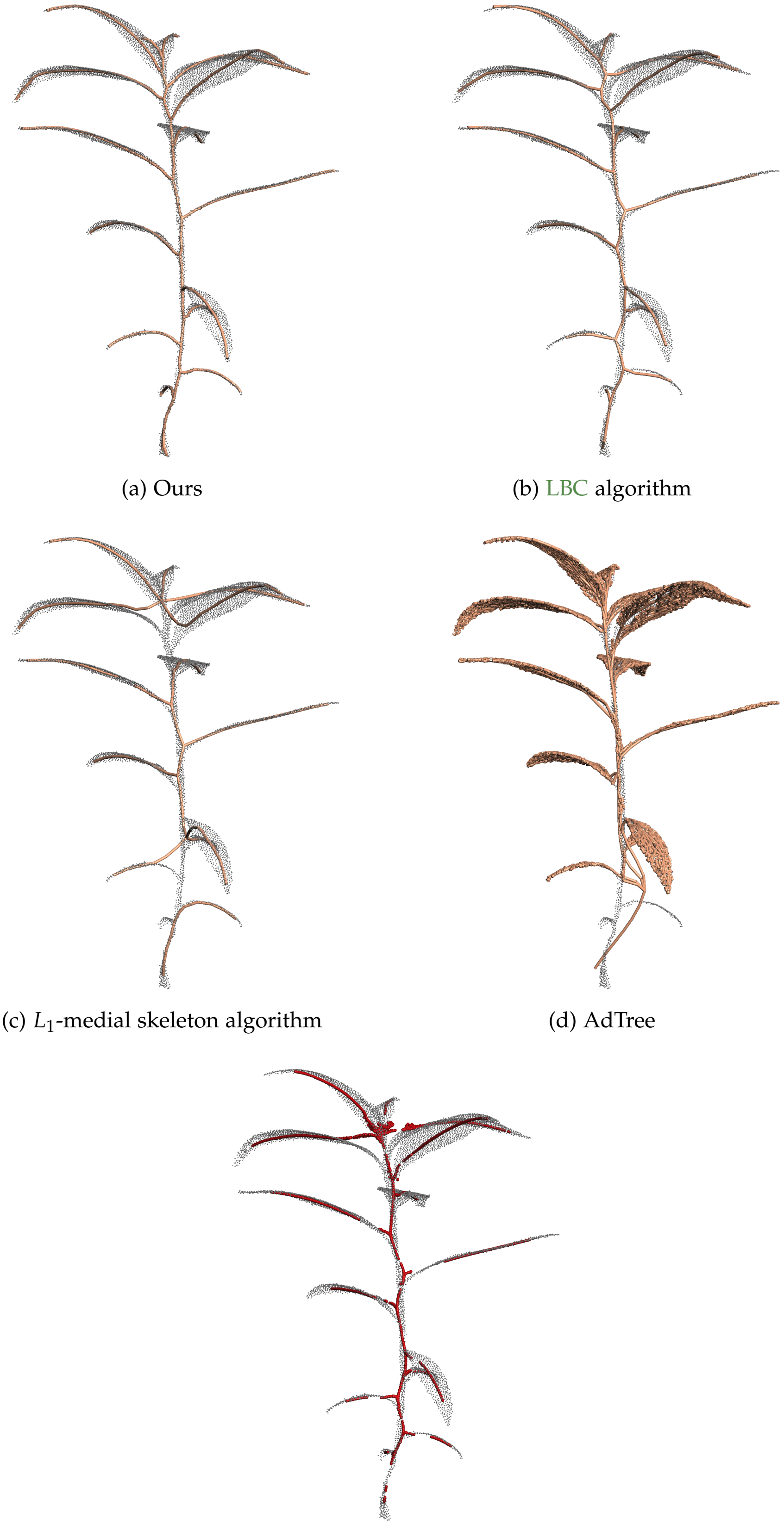
(b) LBC algorithm

(c) L_1-medial skeleton algorithm

(d) AdTree

(e) WoodSKE (Discrete contracted skeleton points)

Figure 4.12: Skeletonization results comparison (Plant C).



(a) Ours

(b) LBC algorithm

(c) L_1-medial skeleton algorithm

(d) AdTree

(e) WoodSKE (Discrete contracted skeleton points)

Figure 4.13: Skeletonization results comparison (Plant D).

According to Figure 4.14, our algorithm consistently extracts satisfactory curve skeletons from various leafy plants. By adjusting the downsample target number, our skeleton extraction algorithm, with default settings, demonstrates generalizability across diverse leafy plant species.

For the four tree point clouds, which contain relatively complex structures and fine details, we downsample the data to $204,800$ points to preserve their geometric information. Given the large input size and intricate structures, the k-nearest neighbors strategy is more suitable for these cases. All other parameters are set to their default values.

Figure 4.15 presents the extracted tree skeletons. The results indicate that the skeletons produced by our algorithm effectively represent the overall geometric shape of the target trees. For tree point clouds with clear branch structures, such as Tree A (Fig. 4.15a) and Tree D (Fig. 4.15d), the quality of extracted skeleton is relatively higher. In contrast, for trees with dense leaf points, the crown geometry characteristics are challenging to capture accurately, resulting in comparatively lower-quality skeletons.

In summary, by appropriately tuning the downsample target number, neighbor searching strategy, and initial parameters (k^0 or rad^0), our skeleton extraction algorithm can effectively handle a wide variety of plant species. This versatility enables the algorithm to be applied across several applications involving different plant types.

4.2.5 Limitations

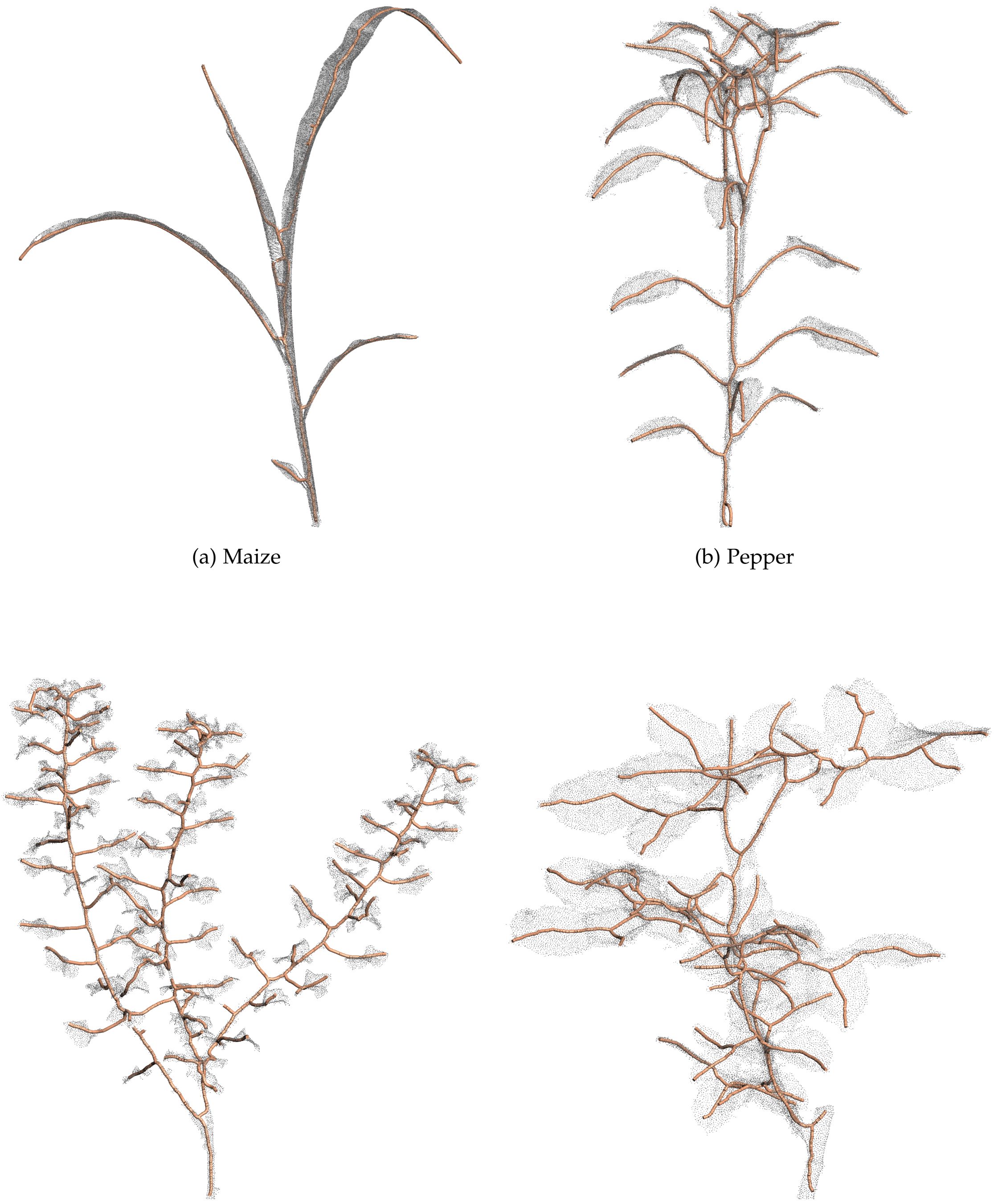
Every algorithm has its limitations, and our proposed framework is no exception. While it effectively extracts curve skeletons from plant point clouds and segments stems and leaves based on the curve skeleton, several limitations still need to be addressed in future work.

First, our plant skeletonization algorithm is geometry-based. The algorithm can accurately extract curve skeletons with correct topological and geometric information for input point clouds with high data quality, particularly those without significant outliers or holes. However, when applied to low-quality point clouds, the results may include abnormal branches or connections, leading to incorrect connectivity and unsatisfactory segmentation in the subsequent steps. This issue can be mitigated in many cases by slightly increasing the initial parameter, k^0 or rad^0. However, this adjustment carries the risk of overlooking small geometric structures. Thus, according to their plant structure and point cloud quality, the user should choose a suitable k^0 or rad^0 carefully.

In addition to the input data quality, another limitation is that our plant skeleton extraction algorithm struggles with some ***P. lapathifolium*** plants with certain structures. The issue arises while building connectivity among extracted skeleton points by using Kruskal's MST on the initial skeleton graph. If two leaves are spatially close and overlap, the algorithm may generate incorrect edges in the initial graph, resulting in topological errors in the final skeleton (Fig. 4.16). A more refined connectivity approach could be developed to address this issue. This method would rely on the extracted skeleton points and incorporate geometric features from nearby regions of the original point cloud. The algorithm could determine whether an edge should be connected or removed by considering differences in geometric features. This refined approach would be a promising direction for future research.

Moreover, we currently treat the skeleton point with the lowest height value as the root vertex in the segmentation component. While this method is sufficient in most cases, it fails when the plant has abnormal growth direction or downward-oriented leaves (see as in Figure 4.17a for an example). Additionally, our segmentation approach only distinguishes between stems and leaves, which is insufficient for practical agricultural applications, where the segmentation of more plant organs, such as flowers (Fig. 4.17b) and fruits, is required. As our framework relies solely on geometric information and does not fully leverage available RGB data, a potential solution for future work would involve combining segmentation results with differences in geometric features and RGB information. This integration could help distinguish more plant organs and improve segmentation accuracy.

Moreover, our skeleton-based segmentation cannot differentiate between the leaf and petiole parts, as both belong to the same skeleton branch and segment in the skeleton representation. Incorporating



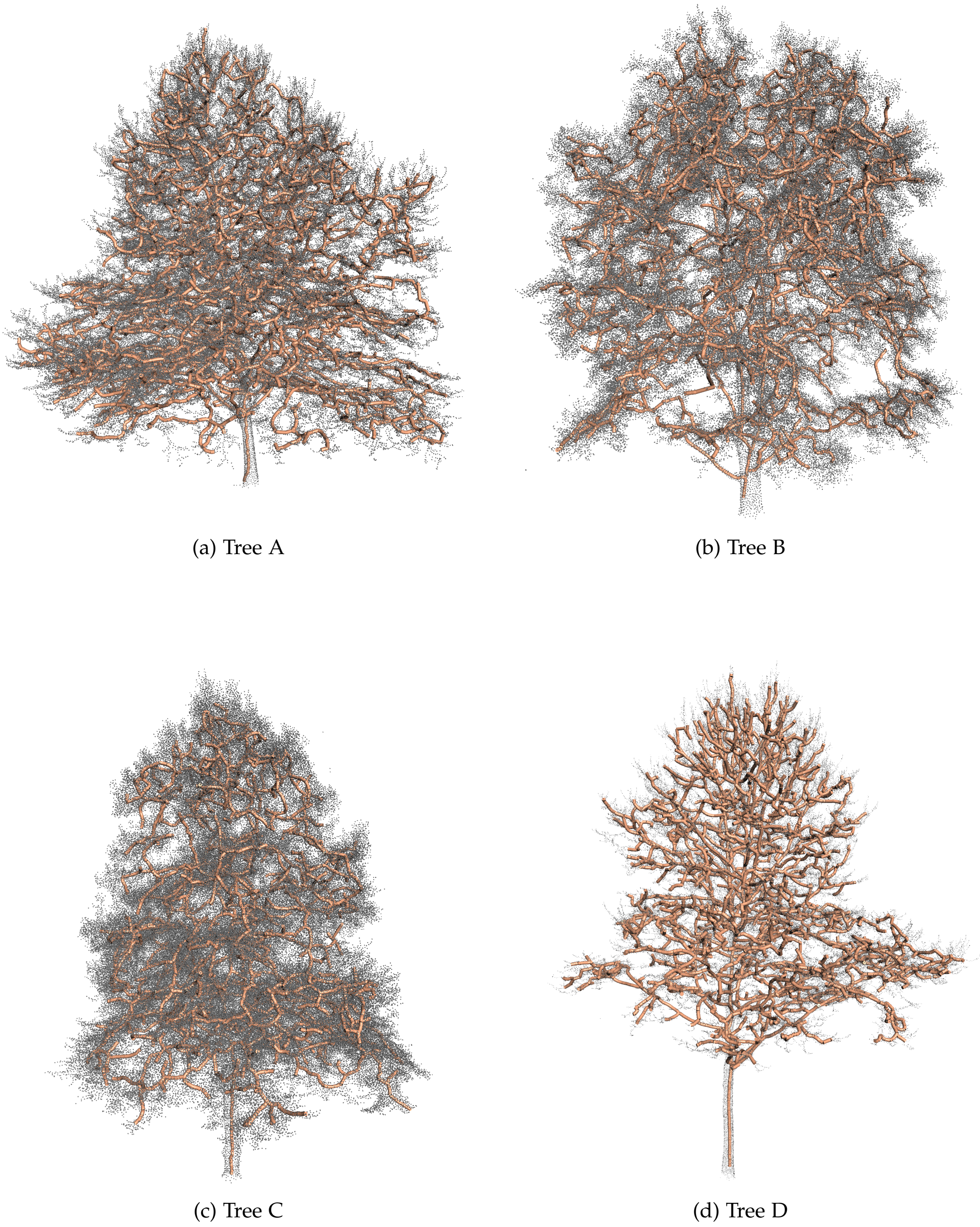
(a) Maize

(b) Pepper

(c) Ribes

(d) Rose

Figure 4.14: Skeletonization results of other leafy plants. Gray points represent the original plant point cloud. The brow cylinders present the skeleton. This color scheme is consistent across the following figures.



(a) Tree A

(b) Tree B

(c) Tree C

(d) Tree D

Figure 4.15: Skeletonization results of trees.

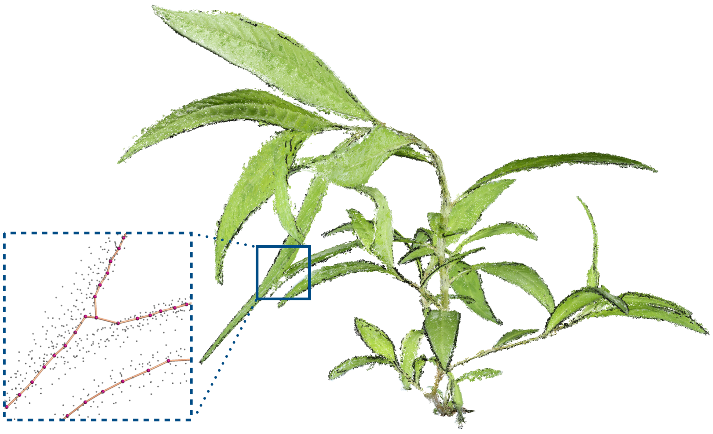
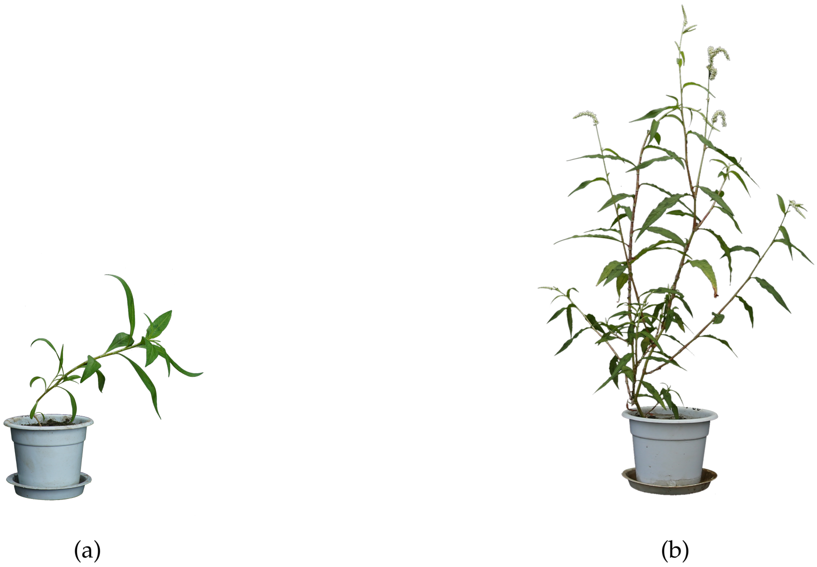


Figure 4.16: Example of a failure case caused by overlapping leaves.



(a)

(b)

Figure 4.17: Two examples that limited our framework. (a) A plant exhibiting abnormal growth direction with downward-oriented leaves. (b) A ***P. lapathifolium*** plant at the mature stage with multiple flowers.

the geometric information from the original point cloud is a potential solution to address this issue. By analyzing the local geometric feature changes along the skeleton segment, the petiole and leaf regions can typically be identified through abrupt changes in these features.

4.2.6 Potential applications

Our proposed framework enables the accurate extraction of plant curve skeletons from point clouds. Beyond facilitating skeleton-based plant organ segmentation, it provides a foundation for phenotyping several essential parameters.

A well-studied area in plant science involves extracting structural characteristics such as plant height, leaf length, branch angle, and leaf angle. Accurate plant skeletons allow researchers to monitor and explore growth patterns and developmental stages by tracking shape or structural changes. Therefore, the curve skeletons extracted by our framework offer the necessary data for users to compute these parameters more explicitly. Moreover, as concise representations encoding both local and global geometric features, skeletons can also be applied in agricultural robotics. Agricultural robots can be provided with efficient reference from those 3D skeleton models that can navigate through crops without causing damage and perform tasks such as pruning [Schneider et al., 2023] or harvesting.

Beyond agricultural field practices, our skeleton extraction algorithm also demonstrates potential in extracting tree skeletons. These tree skeletons can be applied in structural analyses or 3D tree model reconstruction by integrating with the AdTree reconstruction component. Such applications are helpful for developing smart agricultural systems and creating digital twins of individual plants. Digital twins provide valuable tools for simulating and assessing how plants and trees respond to different environmental conditions, such as drought, salinity, shading, or airflow, thereby supporting research on plant development in extreme environments.

5 Conclusion and Future Work

5.1 Conclusion

In this study, we propose a skeleton-based plant organ segmentation framework that can accurately extract curve skeletons from individual point clouds and establish stem-leaf segmentation. It could produce skeletons with both topological and geometric accuracy and achieve satisfactory results overall.

One notable improvement of our framework is its ability to extract precise skeletons from leafy plants, performing well on both leaves and stems. With our Contraction Laplacian Operator, Adaptive Constraint and Tip Points Preservation, our skeletons are able to preserve the fine structure of plants during the skeletonization process. This integration enhances the overall quality and accuracy of the skeleton extraction. Our experimental results shown in Section 4.1 demonstrate that our frame can robustly process various shapes and sizes of plants. As long as the input plant point cloud has a normal growth direction and is of high quality (i.e., no significant holes or noisy points), our framework is able to extract curve skeletons and establish plant stem-leaf segmentation with satisfactory quality.

In summary, the main contributions of this study are as follows:

- **Extension of the LBC algorithm:** We enhance the LBC algorithm for leafy plant skeleton extraction by applying a constrained Laplacian operator, which consists of the Tufted Laplacian operator [Sharp and Crane, 2020] with additional Dynamic Neighbor Searching and Edge Collapse modules. We also introduce Adaptive Constraint and Tip Points Preservation modules into the contraction loops of the LBC algorithm to further refine the skeleton quality.
- **Skeleton Points Calibration:** We adapt the LOP operator [Lipman et al., 2007] as a post-processing step to ensure that the extracted skeleton aligns with the original plant shape and is centrally positioned.
- **3D Plant Point Cloud Dataset:** We introduce a photogrammetric point cloud dataset of plants, comprising 56 *P. lapathifolium* plants with detailed annotations.

And our research questions raised in Section 1.2 can be answered in the following:

- **How can we maintain correct contraction directions while preserving local and global geometric information?**

  The instability of the Laplacian operator L is the primary cause of abnormal contraction directions. By applying the constrained Laplacian operator $\mathcal{L}$, we successfully address this issue, maintaining correct contraction directions while preserving the geometric integrity of both local and global structures.

- **How can we contract a leafy plant point cloud completely into a skeletal object without leaving significant under-contraction areas?**

  By introducing **Adaptive Constraint** modules that apply varying contraction forces based on the directionality degree of each point, we are able to handle both over-contraction and under-contraction scenarios efficiently. This method ensures that the point cloud contracts fully into a line-like object.

- **How can we ensure the extracted skeleton preserves the overall plant shape effectively (i.e., without significant gaps between skeleton points)**

  Our **Distance Constraint** module guarantees the integrity of the extracted skeleton. By setting a maximum allowable distance between points and their neighbors during the contraction process, we are able to reduce gaps in the skeleton, thus ensuring the integrity of the extracted skeleton.

- **How can we ensure that the shape of the extracted skeleton fits the original point cloud well?**

  By using the Contraction Laplacian operator $\mathcal{L}$, point contraction could follow the correct direction, alleviating abnormal contraction issues. Additionally, with the **LOP-based calibration** module, our skeleton is relocated to local L_1 medial positions, ensuring it is centered within the original point cloud and fits the original point cloud well.

- **How effective is our refined LBC algorithm for plant stem-leaf segmentation?**

  With an accurate skeleton that could represent the topological and geometric features of the original plant point cloud, we are enabled to obtain high-quality stem-leaf segmentation results. As shown in Table 4.2 and Table 4.3, the mean precision of semantic segmentation exceeds 84%, while the precision of leaf instance segmentation surpasses 95%.

5.2 Future work

For future work, we would like to continue our studies with a focus on our limitations listed in Section 4.2.5.

- Our current skeleton is generated by computing an MST graph, which is a simple and efficient method in most cases. However, this approach does not involve biological prior knowledge, sometimes leading to incorrect connections. To mitigate this, future work will focus on integrating biological prior knowledge when building connections, particularly to avoid errors caused by overlapping branches or leaves.
- The current segmentation strategy assumes that the skeleton point with the lowest height value corresponds to the root vertex of the plant. While effective in many cases, this assumption fails when dealing with plants exhibiting abnormal growth patterns or downward-facing leaves, which can lead to incorrect identification of plant organs. Future research should explore combining color (RGB) information with geometric features on the original point cloud to improve the accuracy of root vertex identification.
- Additionally, by utilizing differences in color, texture, and shape, the algorithm can be extended to differentiate between various plant organs rather than being limited to stem and leaf classification. The development of deep learning models, which could fuse the information of the original plant point cloud and plant skeleton, could further enhance segmentation performance, especially in complex real-world scenarios.

Bibliography

Au, O. K. C., Tai, C. L., Chu, H. K., Cohen Or, D., and Lee, T. Y. (2008). Skeleton extraction by mesh contraction. *ACM Transactions on Graphics*, 27(3):1–10.

Belkin, M., Sun, J., and Wang, Y. (2008). Discrete laplace operator on meshed surfaces. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, SoCG08. ACM.

Berger, B., Parent, B., and Tester, M. (2010). High-throughput shoot imaging to study drought responses. *Journal of Experimental Botany*, 61(13):3519–3528.

Bucksch, A. (2014). A practical introduction to skeletons for the plant sciences. *Applications in Plant Sciences*, 2(8).

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

Cao, J., Tagliasacchi, A., Olson, M., Zhang, H., and Su, Z. (2010). Point cloud skeletons via laplacian based contraction. In *2010 Shape Modeling International Conference*. IEEE.

Chaivivatrakul, S., Tang, L., Dailey, M. N., and Nakarmi, A. D. (2014). Automatic morphological trait characterization for corn plants via 3d holographic reconstruction. *Computers and Electronics in Agriculture*, 109:109–123.

Chaudhury, A. and Godin, C. (2020). Skeletonization of plant point cloud data using stochastic optimization framework. *Frontiers in Plant Science*, 11.

Crane, K., Weischedel, C., and Wardetzky, M. (2017). The heat method for distance computation. *Commun. ACM*, 60(11):90–99.

Das Choudhury, S., Bashyam, S., Qiu, Y., Samal, A., and Awada, T. (2018). Holistic and component plant phenotyping using temporal image sequence. *Plant Methods*, 14(1).

Das Choudhury, S., Samal, A., and Awada, T. (2019). Leveraging image analysis for high-throughput plant phenotyping. *Frontiers in Plant Science*, 10.

Deery, D. M., Rebetzke, G. J., Jimenez-Berni, J. A., James, R. A., Condon, A. G., Bovill, W. D., Hutchinson, P., Scarrow, J., Davy, R., and Furbank, R. T. (2016). Methodology for high-throughput field phenotyping of canopy temperature using airborne thermography. *Frontiers in Plant Science*, 7.

Desbrun, M., Meyer, M., Schröder, P., and Barr, A. H. (1999). Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99*, SIGGRAPH '99, page 317–324. ACM Press.

Du, R., Ma, Z., Xie, P., He, Y., and Cen, H. (2023). Pst: Plant segmentation transformer for 3d point clouds of rapeseed plants at the podding stage. *ISPRS Journal of Photogrammetry and Remote Sensing*, 195:380–392.

Du, S., Lindenbergh, R., Ledoux, H., Stoter, J., and Nan, L. (2019). Adtree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing*, 11(18):2074.

Eldar, Y., Lindenbaum, M., Porat, M., and Zeevi, Y. (1997). The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing*, 6(9):1305–1315.

Elnashef, B., Filin, S., and Lati, R. N. (2019). Tensor-based classification and segmentation of three-dimensional point clouds for organ-level plant phenotyping and growth analysis. *Computers and Electronics in Agriculture*, 156:51–61.

Fan, L., Pang, Z., Zhang, T., Wang, Y.-X., Zhao, H., Wang, F., Wang, N., and Zhang, Z. (2022). Embracing single stride 3d object detector with sparse transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8458–8468.

Forero, M. G., Murcia, H. F., Méndez, D., and Betancourt-Lozano, J. (2022). Lidar platform for acquisition of 3d plant phenotyping database. *Plants*, 11(17):2199.

Gelard, W., Devy, M., Herbulot, A., and Burger, P. (2017). Model-based segmentation of 3d point clouds for phenotyping sunflower plants. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications.

Ghahremani, M., Williams, K., Corke, F., Tiddeman, B., Liu, Y., Wang, X., and Doonan, J. H. (2021). Direct and accurate feature extraction from 3d point clouds of plants using ransac. *Computers and Electronics in Agriculture*, 187:106240.

Guennebaud, G., Jacob, B., et al. (2010). Eigen v3. http://eigen.tuxfamily.org.

Guo, M.-H., Cai, J.-X., Liu, Z.-N., Mu, T.-J., Martin, R. R., and Hu, S.-M. (2021a). Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., and Bennamoun, M. (2021b). Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364.

Harandi, N., Vandenberghe, B., Vankerschaver, J., Depuydt, S., and Van Messem, A. (2023). How to make sense of 3d representations for plant phenotyping: a compendium of processing and analysis techniques. *Plant Methods*, 19(1).

Hildebrandt, K. and Polthier, K. (2011). On approximation of the laplace–beltrami operator and the willmore energy of surfaces. *Computer Graphics Forum*, 30(5):1513–1520.

Hu, K., Ying, W., Pan, Y., Kang, H., and Chen, C. (2024). High-fidelity 3d reconstruction of plants using neural radiance fields. *Computers and Electronics in Agriculture*, 220:108848.

Hu, T., Chitnis, N., Monos, D., and Dinh, A. (2021). Next-generation sequencing technologies: An overview. *Human Immunology*, 82(11):801–811.

Huang, H., Wu, S., Cohen-Or, D., Gong, M., Zhang, H., Li, G., and Chen, B. (2013). L 1 -medial skeleton of point cloud. *ACM Transactions on Graphics*, 32(4):1–8.

Jiang, H., Jang, J., and Kpotufe, S. (2018). Quickshift++: Provably good initializations for sample-based mean shift. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2294–2303. PMLR.

Jiang, L., Zhao, H., Shi, S., Liu, S., Fu, C.-W., and Jia, J. (2020). Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jimenez-Berni, J. A., Deery, D. M., Rozas-Larraondo, P., Condon, A. T. G., Rebetzke, G. J., James, R. A., Bovill, W. D., Furbank, R. T., and Sirault, X. R. R. (2018). High throughput determination of plant height, ground cover, and above-ground biomass in wheat with lidar. *Frontiers in Plant Science*, 9.

Jin, S., Su, Y., Wu, F., Pang, S., Gao, S., Hu, T., Liu, J., and Guo, Q. (2019). Stem–leaf segmentation and phenotypic trait extraction of individual maize using terrestrial lidar data. *IEEE Transactions on Geoscience and Remote Sensing*, 57(3):1336–1346.

Jin, S., Sun, X., Wu, F., Su, Y., Li, Y., Song, S., Xu, K., Ma, Q., Baret, F., Jiang, D., Ding, Y., and Guo, Q. (2021). Lidar sheds new light on plant phenomics for plant breeding and management: Recent advances and future prospects. *ISPRS Journal of Photogrammetry and Remote Sensing*, 171:202–223.

Krishna, T. P. A., Veeramuthu, D., Maharajan, T., and Soosaimanickam, M. (2023). The era of plant breeding: Conventional breeding to genomics-assisted breeding for crop improvement. *Current Genomics*, 24(1):24–35.

Li, C., Zhou, M., Geng, G., Xie, Y., Zhang, Y., and Liu, Y. (2023). Epcs: Endpoint-based part-aware curve skeleton extraction for low-quality point clouds. *Computers & Graphics*, 117:209–221.

Li, D., Li, J., Xiang, S., and Pan, A. (2022a). Psegnet: Simultaneous semantic and instance segmentation for point clouds of plants. *Plant Phenomics*, 2022.

Li, D., Shi, G., Li, J., Chen, Y., Zhang, S., Xiang, S., and Jin, S. (2022b). Plantnet: A dual-function point cloud segmentation network for multiple plant species. *ISPRS Journal of Photogrammetry and Remote Sensing*, 184:243–263.

Li, L. and Wang, W. (2018). Improved use of lop for curve skeleton extraction. *Computer Graphics Forum*, 37(7):313–323.

Li, Y., Su, Y., Zhao, X., Yang, M., Hu, T., Zhang, J., Liu, J., Liu, M., and Guo, Q. (2020). Retrieval of tree branch architecture attributes from terrestrial laser scan data using a laplacian algorithm. *Agricultural and Forest Meteorology*, 284:107874.

Li, Y., Wen, W., Miao, T., Wu, S., Yu, Z., Wang, X., Guo, X., and Zhao, C. (2022c). Automatic organ-level point cloud segmentation of maize shoots by integrating high-throughput data acquisition and deep learning. *Computers and Electronics in Agriculture*, 193:106702.

Lipman, Y., Cohen-Or, D., Levin, D., and Tal-Ezer, H. (2007). Parameterization-free projection for geometry reconstruction. *ACM Transactions on Graphics*, 26(3):22.

Liu, H. and Yan, J. (2018). Crop genome-wide association study: a harvest of biological relevance. *The Plant Journal*, 97(1):8–18.

Liu, Y., Zhang, G., Shao, K., Xiao, S., Wang, Q., Zhu, J., Wang, R., Meng, L., and Ma, Y. (2022). Segmentation of individual leaves of field grown sugar beet plant based on 3d point cloud. *Agronomy*, 12(4):893.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022.

Ma, Z., Du, R., Xie, J., Sun, D., Fang, H., Jiang, L., and Cen, H. (2023). Phenotyping of silique morphology in oilseed rape using skeletonization with hierarchical segmentation. *Plant Phenomics*, 5.

Ma, Z., Sun, D., and Cen, H. (2021a). A novel skeletonization algorithm combined with hierarchical segmentation for phenotyping siliques of oilseed rape. In *2021 ASABE Annual International Virtual Meeting, July 12-16, 2021*, virt2021. American Society of Agricultural and Biological Engineers.

Ma, Z., Sun, D., Xu, H., Zhu, Y., He, Y., and Cen, H. (2021b). Optimization of 3d point clouds of oilseed rape plants based on time-of-flight cameras. *Sensors*, 21(2):664.

Mertoğlu, K., Şalk, Y., Sarıkaya, S. K., Turgut, K., Evrenesoğlu, Y., undefinedevikalp, H., Gerek, O. N., Dutağacı, H., and Rousseau, D. (2024). Planest-3d: A new annotated dataset for segmentation of 3d plant point clouds.

Meyer, L., Gilson, A., Schmidt, U., and Stamminger, M. (2024). Fruitnerf: A unified neural radiance field based fruit counting framework.

Meyer, M., Desbrun, M., Schröder, P., and Barr, A. H. (2003). *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, page 35–57. Springer Berlin Heidelberg.

Miao, T., Zhu, C., Xu, T., Yang, T., Li, N., Zhou, Y., and Deng, H. (2021). Automatic stem-leaf segmentation of maize shoots using three-dimensional point cloud. *Computers and Electronics in Agriculture*, 187:106310.

Mir, R. R., Reynolds, M., Pinto, F., Khan, M. A., and Bhat, M. A. (2019). High-throughput phenotyping for crop improvement in the genomics era. *Plant Science*, 282:60–72.

Nan, L. (2021). Easy3d: a lightweight, easy-to-use, and efficient c++ library for processing and rendering 3d data. *Journal of Open Source Software*, 6(64):3255.

Pape, J.-M. and Klukas, C. (2015). *3-D Histogram-Based Segmentation and Leaf Detection for Rosette Plants*, page 61–74. Springer International Publishing.

Paproki, A., Sirault, X., Berry, S., Furbank, R., and Fripp, J. (2012). A novel mesh processing based technique for 3d plant analysis. *BMC Plant Biology*, 12(1):63.

Patel, A. K., Park, E.-S., Lee, H., Priya, G. G. L., Kim, H., Joshi, R., Arief, M. A. A., Kim, M. S., Baek, I., and Cho, B.-K. (2023). Deep learning-based plant organ segmentation and phenotyping of sorghum plants using lidar point cloud. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 16:8492–8507.

Paulus, S. (2019). Measuring crops in 3d: using geometry for plant phenotyping. *Plant Methods*, 15(1).

Paulus, S., Dupuis, J., Mahlein, A.-K., and Kuhlmann, H. (2013). Surface feature based classification of plant organs from 3d laserscanned point clouds for plant phenotyping. *BMC Bioinformatics*, 14(1).

Praveen Kumar, J. and Domnic, S. (2019). Image based leaf segmentation and counting in rosette plants. *Information Processing in Agriculture*, 6(2):233–246.

Qi, C. R., Su, H., Kaichun, M., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Raykov, Y. P., Boukouvalas, A., Baig, F., and Little, M. A. (2016). What to do when k-means clustering fails: A simple yet principled alternative algorithm. *PLOS ONE*, 11(9):e0162259.

Ren, M. and Zemel, R. S. (2017). End-to-end instance segmentation with recurrent attention. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Rose, J., Paulus, S., and Kuhlmann, H. (2015). Accuracy analysis of a multi-view stereo approach for phenotyping of tomato plants at the organ level. *Sensors*, 15(5):9651–9665.

Scharr, H., Minervini, M., French, A. P., Klukas, C., Kramer, D. M., Liu, X., Luengo, I., Pape, J.-M., Polder, G., Vukadinovic, D., Yin, X., and Tsaftaris, S. A. (2015). Leaf segmentation in plant phenotyping: a collation study. *Machine Vision and Applications*, 27(4):585–606.

Schneider, E., Jayanth, S., Silwal, A., and Kantor, G. (2023). 3d skeletonization of complex grapevines for robotic pruning. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume abs/1411.4038, page 3278–3283. IEEE.

Schunck, D., Magistri, F., Rosu, R. A., Cornelißen, A., Chebrolu, N., Paulus, S., Léon, J., Behnke, S., Stachniss, C., Kuhlmann, H., and Klingbeil, L. (2021). Pheno4d: A spatio-temporal dataset of maize and tomato plant point clouds for phenotyping and advanced plant analysis. *PLOS ONE*, 16(8):e0256340.

Sharp, N. and Crane, K. (2020). A laplacian for nonmanifold triangle meshes. *Computer Graphics Forum*, 39(5):69–80.

Sharp, N., Crane, K., et al. (2019a). Geometrycentral: A modern c++ library of data structures and algorithms for geometry processing. `https://geometry-central.net/`.

Sharp, N. et al. (2019b). Polyscope. www.polyscope.run.

Shi, W., van de Zedde, R., Jiang, H., and Kootstra, G. (2019). Plant-part segmentation using deep learning and multi-view vision. *Biosystems Engineering*, 187:81–95.

Siek, J. G., Lee, L.-Q., and Lumsdaine, A. (2001). *The Boost Graph Library: User Guide and Reference Manual, The*. Pearson Education.

Su, Z., Li, S., Liu, H., and He, Z. (2019). Tree skeleton extraction from laser scanned points. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE.

Sun, S., Li, C., Chee, P. W., Paterson, A. H., Jiang, Y., Xu, R., Robertson, J. S., Adhikari, J., and Shehzad, T. (2020). Three-dimensional photogrammetric mapping of cotton bolls in situ based on point cloud segmentation and clustering. *ISPRS Journal of Photogrammetry and Remote Sensing*, 160:195–207.

Sun, S., Li, C., Chee, P. W., Paterson, A. H., Meng, C., Zhang, J., Ma, P., Robertson, J. S., and Adhikari, J. (2021). High resolution 3d terrestrial lidar for cotton plant main stalk and node detection. *Computers and Electronics in Agriculture*, 187:106276.

Tagliasacchi, A., Delame, T., Spagnuolo, M., Amenta, N., and Telea, A. (2016). 3d skeletons: A state-of-the-art report. *Computer Graphics Forum*, 35(2):573–597.

Tardieu, F., Cabrera-Bosquet, L., Pridmore, T., and Bennett, M. (2017). Plant phenomics, from sensors to knowledge. *Current Biology*, 27(15):R770–R783.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wahabzada, M., Paulus, S., Kersting, K., and Mahlein, A.-K. (2015). Automated interpretation of 3d laserscanned point clouds for plant organ segmentation. *BMC Bioinformatics*, 16(1).

Walter, A., Liebisch, F., and Hund, A. (2015). Plant phenotyping: from bean weighing to image analysis. *Plant Methods*, 11(1):14.

Wang, D., Song, Z., Miao, T., Zhu, C., Yang, X., Yang, T., Zhou, Y., Den, H., and Xu, T. (2023). Dfsp: A fast and automatic distance field-based stem-leaf segmentation pipeline for point cloud of maize shoot. *Frontiers in Plant Science*, 14.

Wang, X., Zhou, Y., Pan, X., and Zhang, C. (2017). A robust 3d point cloud skeleton extraction method. *SCIENTIA SINICA Informationis*, 47(7):832.

Wang, Y., Liu, Q., Yang, J., Ren, G., Wang, W., Zhang, W., and Li, F. (2024). A method for tomato plant stem and leaf segmentation and phenotypic extraction based on skeleton extraction and supervoxel clustering. *Agronomy*, 14(1):198.

Wen, Q., Tafrishi, S. A., Ji, Z., and Lai, Y.-K. (2024). Glskeleton: A geometric laplacian-based skeletonisation framework for object point clouds. *IEEE Robotics and Automation Letters*, 9(5):4615–4622.

Weng, Y., Xu, W., Wu, Y., Zhou, K., and Guo, B. (2006). 2d shape deformation using nonlinear least squares optimization. *The Visual Computer*, 22(9–11):653–660.

Wu, B., Zheng, G., Chen, Y., and Yu, D. (2021). Assessing inclination angles of tree branches from terrestrial laser scan data using a skeleton extraction method. *International Journal of Applied Earth Observation and Geoinformation*, 104:102589.

Wu, S., Wen, W., Xiao, B., Guo, X., Du, J., Wang, C., and Wang, Y. (2019). An accurate skeleton extraction approach from 3d point clouds of maize plants. *Frontiers in Plant Science*, 10.

Xiang, L., Bao, Y., Tang, L., Ortiz, D., and Salas-Fernandez, M. G. (2019). Automated morphological traits extraction for sorghum plants via 3d point cloud data analysis. *Computers and Electronics in Agriculture*, 162:951–961.

Xiao, Q., Bai, X., Zhang, C., and He, Y. (2022). Advanced high-throughput plant phenotyping techniques for genome-wide association studies: A review. *Journal of Advanced Research*, 35:215–230.

Xiao, Y., Liu, H., Wu, L., Warburton, M., and Yan, J. (2017). Genome-wide association studies in maize: Praise and stargaze. *Molecular Plant*, 10(3):359–374.

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

Xu, M., Ding, R., Zhao, H., and Qi, X. (2021). Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3173–3182.

Xu, N., Sun, G., Bai, Y., Zhou, X., Cai, J., and Huang, Y. (2023). Global reconstruction method of maize population at seedling stage based on kinect sensor. *Agriculture*, 13(2):348.

Yang, D., Yang, H., Liu, D., and Wang, X. (2024). Research on automatic 3d reconstruction of plant phenotype based on multi-view images. *Computers and Electronics in Agriculture*, 220:108866.

Yang, W., Feng, H., Zhang, X., Zhang, J., Doonan, J. H., Batchelor, W. D., Xiong, L., and Yan, J. (2020). Crop phenomics and high-throughput phenotyping: Past decades, current challenges, and future perspectives. *Molecular Plant*, 13(2):187–214.

Yazdi, H., Shu, Q., Rötzer, T., Petzold, F., and Ludwig, F. (2024). A multilayered urban tree dataset of point clouds, quantitative structure and graph models. *Scientific Data*, 11(1).

Yin, X., Liu, X., Chen, J., and Kramer, D. M. (2018). Joint multi-leaf segmentation, alignment, and tracking for fluorescence plant videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1411–1423.

Zhang, H., Wang, L., Jin, X., Bian, L., and Ge, Y. (2023). High-throughput phenotyping of plant leaf morphological, physiological, and biochemical traits on multiple scales using optical sensing. *The Crop Journal*, 11(5):1303–1318.

Zhang, K., Hao, M., Wang, J., Chen, X., Leng, Y., de Silva, C. W., and Fu, C. (2021). Linked dynamic graph cnn: Learning through point cloud by linking hierarchical features. In *2021 27th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE.

Zhang, X., Huang, C., Wu, D., Qiao, F., Li, W., Duan, L., Wang, K., Xiao, Y., Chen, G., Liu, Q., Xiong, L., Yang, W., and Yan, J. (2017). High-throughput phenotyping and qtl mapping reveals the genetic architecture of maize plant growth. *Plant Physiology*, 173(3):1554–1564.

Zhao, H., Jiang, L., Jia, J., Torr, P. H., and Koltun, V. (2021). Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16259–16268.

Zhou, H., Zhou, Y., Long, W., Wang, B., Zhou, Z., and Chen, Y. (2023). A fast phenotype approach of 3d point clouds of pinus massoniana seedlings. *Frontiers in Plant Science*, 14.

Zhou, Q.-Y., Park, J., and Koltun, V. (2018). Open3d: A modern library for 3d data processing.

Zhou, Y., Sun, P., Zhang, Y., Anguelov, D., Gao, J., Ouyang, T., Guo, J., Ngiam, J., and Vasudevan, V. (2020). End-to-end multi-view fusion for 3d object detection in lidar point clouds. In Kaelbling, L. P., Kragic, D., and Sugiura, K., editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 923–932. PMLR.

Zhou, Y. and Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Palatino.

TU Delft