

Tree Skeletonization from 3D Point Clouds by Denoising Diffusion

Elias Ariel Marks^{1,2} Lucas Nunes^{1,2} Federico Magistri^{1,2} Matteo Sodano^{1,2}
 Rodrigo Marcuzzi^{1,2} Lars Zimmermann² Jens Behley^{1,2} Cyrill Stachniss^{1,2,3}

¹Center for Robotics ²University of Bonn ³Lamarr Institute for Machine Learning and Artificial Intelligence

{firstname.lastname}@igg.uni-bonn.de

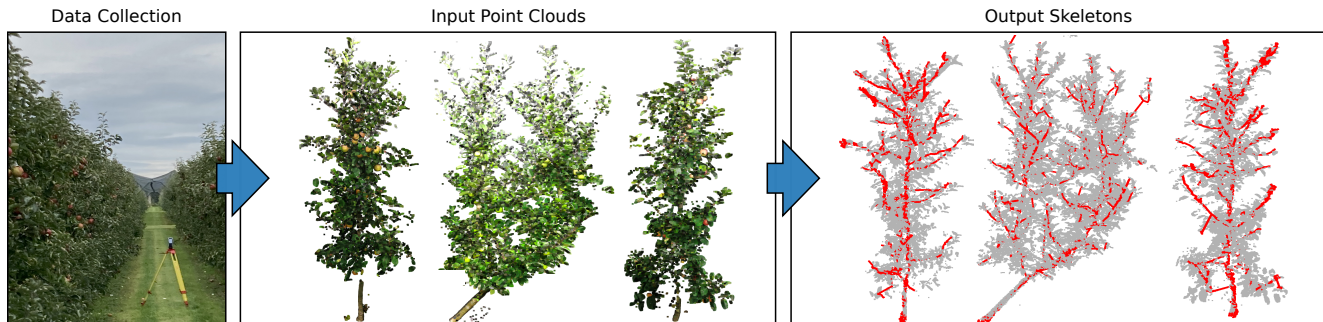


Figure 1. Our approach generates tree skeletons (right) of real orchard trees (left) from colorized point clouds (middle). We leverage a denoising diffusion probabilistic model to predict the nodes and hierarchy of tree skeleton represented as a graph (red) of individual trees.

Abstract

The natural world presents complex organic structures, such as tree canopies, that humans can interpret even when only partially visible. Understanding tree structures is key for forest monitoring, orchard management, and automated harvesting applications. However, reconstructing tree topologies from sensor data, called tree skeletonization, remains a challenge for computer vision approaches. Traditional methods for tree skeletonization rely on hand-crafted features, regression, or generative models, whereas recent advances focus on deep learning approaches. Existing methods often struggle with occlusions caused by dense foliage, limiting their applicability over the annual vegetation cycle. Furthermore, the lack of real-world data with reference information limits the evaluation of these methods to synthetic datasets, which does not validate generalization to real environments. In this paper, we present a novel approach for tree skeletonization that combines a generative denoising diffusion probabilistic model for predicting node positions and branch directions with a classical minimum spanning tree algorithm to infer tree skeletons from 3D point clouds, even with strong occlusions. Additionally, we provide a dataset of an apple orchard with 280 trees scanned 10 times during the growing season with corresponding reference skeletons, enabling quantitative evaluation. Experiments show the superior performance of our approach on real-world data and competitive results compared to state-of-art approaches on synthetic benchmarks.

1. Introduction

The world around us is filled with natural structures, such as trees, that humans can interpret even when parts of them are occluded; however, this remains a challenge for computer vision systems. Reconstructing tree topologies from sensor data is one application where this aspect matters. There are many applications where inferring these topologies is fundamental, e.g., forest monitoring for understanding of these complex natural systems [67], monitoring growth [16], predicting quality and quantity of harvested products [14], or carbon storage estimation [9]. Another highly relevant application is orchard management, where automated pruning and thinning processes require an accurate representation of tree topology to define target points [71]. Automated harvesting robots can benefit from topological tree models by including them in the end effector path planning to avoid collisions with trunks and thick branches while ignoring foliage and small twigs [2]. There is an increasing interest in tree topology estimation, which is commonly called tree skeletonization [1, 7, 8, 26, 32, 34, 38]. Tree skeletonization consists of inferring from sensor data the graph representing the medial axes of the trunk, branches, and twigs, collectively referred to as branches, for simplicity.

We address tree skeletonization using 3D point cloud data, which is more descriptive for geometry than using only 2D images. Traditionally, the task of tree skeletonization is approached as a regression problem [5, 6, 65], while more recent approaches tackle the problem also with gen-

erative methods [66]. Prior work is based on traditional optimization strategies, and more recently, the research direction focuses more on the use of deep learning models [13, 37]. These approaches often predict offsets to estimate the medial axis of the branches. This works well when most of the branches are measured by the sensor, but performance degrades when large amounts of leaves lead to heavy self-occlusions of the trees. In this setting of occluded views, generative models have the potential to infer the missing parts by learning the underlying data distribution. While the use of traditional generative methods in the form of procedural tree models has been explored by Stava *et al.* [62], we are not aware of any work employing generative deep learning models for tree skeletonization. We leverage generative denoising diffusion models motivated by their excellent results for generation of image and point cloud data [22, 44, 72].

One major limiting factor in the development of tree skeletonization methods is the lack of real-world reference data for evaluating developed methods. Manual annotation of real-world data is complex; therefore, most existing work is evaluated quantitatively on synthetic data and provides only qualitative results for real-world data. While qualitative results can indicate performance, quantitative metrics are essential to compare different methods objectively. To fill this gap, we recorded real-world 3D point cloud data and provide reference tree skeletons leveraging multiple scans of trees in an orchard over the whole vegetation cycle.

In this paper, we propose a method for estimating skeletons from 3D point cloud data of trees with dense foliage, leveraging a denoising diffusion probabilistic model (DDPM). Our approach takes as input a 3D point cloud that partially covers the tree and outputs a graph representing the branching structure of the trunk and canopy. The DDPM predicts the position of each node and the branching direction to encode connectivity implicitly. We then use these predictions to create a graph of the whole tree from the tips of the branches to the base of the tree using a minimum spanning tree with edge weights based on flow directions. We also provide a point cloud dataset of an apple orchard covering 280 trees measured 10 times over the growing season. We provide reference skeletons of the branching structure for each tree and date, which allows evaluation at different occlusion levels due to leaf growth. Fig. 1 shows the data and exemplary extracted skeletons.

In summary, our key contributions are:

- A tree skeletonization approach using 3D point clouds as input that employs a novel diffusion-based formulation for generating an implicit representation of a graph-like structure from which we reconstruct the partially occluded branching structure of a tree with foliage.
- A novel 3D point cloud dataset covering apple trees in a real orchard with reference skeletons for quantitative

evaluation on real-world data.

- Quantitative evaluation on our real-world dataset showing improved tree skeletonization performance of our approach over other state-of-the-art approaches, but also competitive performance on existing synthetic datasets.

We provide the implementation of our approach as open source code at <https://github.com/PRBonn/DiffTS>.

2. Related Work

Tree skeletonization is a challenging problem in computer vision and has been extensively studied in the past. Previous works for tree skeletonization [1, 7, 32, 38] use 2D images as input, which are convenient to acquire but lack depth information required to reconstruct 3D tree skeletons. This leads to poor performance for complex tree skeletons or situations with dense foliage due to low observability using a single view. Therefore, other methods [8, 26, 34] use 3D point clouds or other means of 3D data to extract the underlying tree skeletons and can be mainly subdivided into (i) geometric and (ii) procedural modeling approaches.

Geometric approaches use different strategies to infer the branch points from the measured 3D point cloud given. Iterative thinning techniques are often used to infer the medial axis of the branches but often fail with incomplete or noisy input data, especially when dense foliage is present [5, 20, 63, 65]. Other approaches use clustering of the input points into branch pieces and connecting them based on geodesic graphs [11, 17, 18, 23, 37, 64], which is efficient but insufficient with occlusions and tight branching structures. As geometric approaches are affected by occlusions leading to missing parts and disconnected branches in the output, some works ensure global connectivity of predicted skeleton by constructing minimum spanning trees [15, 40, 75] or global graph contraction [25].

Procedural approaches infer the branching structure based on apriori knowledge about tree growth [4, 19, 53, 77], which is formalized using learned rules [31] or L-systems [35] that are a rewriting system guided by a set of rules describing the growth of a given tree variety. This leads to a procedural generation of the tree skeleton, starting from the trunk’s base to the twigs’ tip. Zhang *et al.* [74] propose a procedural generation moving in the opposite direction, emulating the path of sap particles from the leaves to the root for estimating the branching structure. Initial particle positions are defined based on point density, and particle directions are computed based on the neighboring particles and the location of the tree base. Overall, procedural approaches perform well on data with missing parts but rely on extensive hyperparameter tuning to generate these missing parts correctly.

Deep learning approaches recently received increasing interest for tree skeletonization. Liu *et al.* [37] predict semantic branching points and branch instances approximated

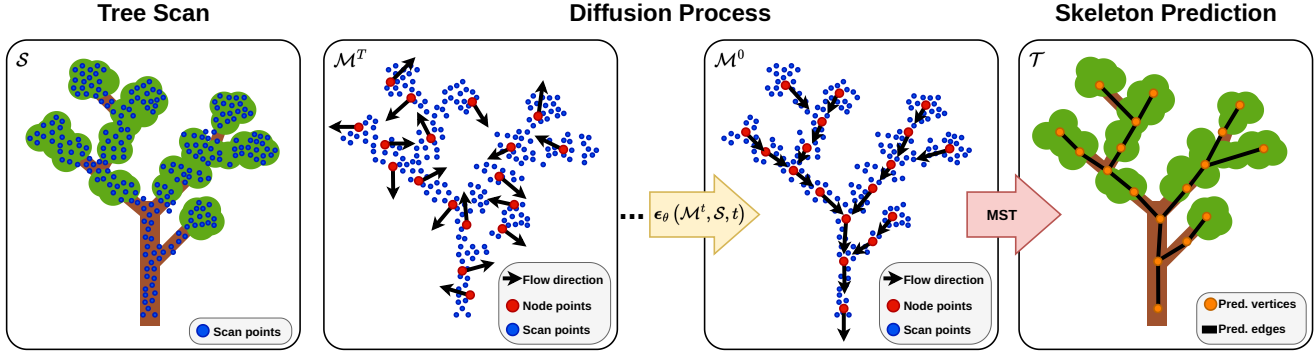


Figure 2. **Overview of our tree skeletonization approach.** Given a tree and its point cloud scan \mathcal{S} , we define the initial set of nodes \mathcal{M}^T with the node points \mathbf{p}_d^T as a set of points sampled from \mathcal{S} and the nodes flow directions \mathbf{f}_d^T randomly initialized. Then, the trained DDPM is used to iteratively predict the noise ϵ_θ at step t from the set of nodes \mathcal{M}^t , conditioned to the scan \mathcal{S} , until arriving at \mathcal{M}^0 . From \mathcal{M}^0 , we compute the minimum spanning tree (MST) to define the predicted tree skeleton \mathcal{T} with vertices \mathcal{V}' and edges \mathcal{E}' .

by cylinders to then apply a neural merging module for predicting the final skeleton. Liu *et al.* [36] infer a graph representing the tree skeleton from a single input image by combining learning-based graph generation with minimum spanning trees. Li *et al.* [33] also use single view images but generate a 3D semantic voxel grid to encode structural information and intricate tree features. Isokane *et al.* [24] instead work with multi-view images to infer parts of the branching structure occluded by leaves and leverage a Bayesian extension of image-to-image translation. Liu *et al.* [39] generate complex 3D tree skeletons from 2D sketches. Meyer *et al.* [48] propose an approach that first predicts which points are part of the main trunk or branches, which is then used for a Laplacian-based contraction to extrapolate the tree skeleton of leaf-free trees.

Denosing diffusion probabilistic models lately became popular due to their high-quality image generation results [12, 22, 50, 52, 54, 56, 78, 79]. Also, conditioned diffusion models gained even more attention due to their ability to steer the generation process towards a specific input [3, 21, 73]. The main drawback of DDPMs is the time needed for the denoising process, which can be reduced, for instance, by distillation of the denoising model [47, 57] or analytically approximating the denoising step solution to reduce the number of needed steps [27, 42, 43, 60].

Diffusion models for 3D data have been investigated for 3D shape generation [44, 45, 58, 59, 69, 72, 76] focusing on point clouds of single objects to generate new shapes or complete partial inputs. Only a few works [30, 49, 80] instead target real-world data generation. Some works [49, 80] rely on projecting the 3D data to an image-based representation, *e.g.*, range images, such that the image-based methods can be directly applied. However, such projection-based approaches often cannot complete the whole 3D scene when reprojecting the image to the 3D world since some regions do not have any information due to occlusions in the projected point cloud. Lee *et al.* [30] achieve

the generation of complete scenes using a discrete diffusion model formulation and a fixed voxel grid representation of the environment. The model is then used to infer whether each voxel is occupied, and a semantic label is predicted. The work by Nunes *et al.* [51] instead operates directly at point level to generate complete 3D scenes. In our work, we leverage this local formulation by Nunes *et al.* [51] to predict the position of the nodes and their flow direction to implicitly represent the tree structure, which we then use to extract the full tree skeleton as a graph.

3D tree skeletonization datasets for evaluation of tree skeletonization approaches are either of synthetic nature or do not provide any means of quantitative evaluation due to the difficulty of obtaining reference skeletons for real data. Dobbs *et al.* [13] present a synthetic dataset of trees of six different species with reference skeletons consisting of only 100 trees per species. The synthetic dataset proposed by Tang *et al.* [68] contains 13,000 trees of 10 different species but only contains the node position of a reference skeleton without any connectivity information. Yan *et al.* [70] also propose a synthetic dataset of trees with sparse point clouds with only around 3,000 points per tree. Meyer *et al.* [48] provide qualitative experiments on real-world data but lack quantitative evaluation due to missing reference skeleton.

Together with our tree skeletonization approach, we present the first real-world dataset of tree point clouds with reference skeletons of 280 trees throughout a vegetative cycle that enables quantitative evaluation of tree skeletonization with different levels of occlusions caused by foliage.

3. Our Approach

In this paper, we aim at extracting the branching structure of a tree, *i.e.* tree skeleton, from the 3D point cloud of a whole tree including the foliage. To this end, we propose a tree skeletonization method based on a denoising diffusion probabilistic model (DDPM) that outputs a tree skeleton based on an input 3D point cloud. In this work, we

define the tree skeleton as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ defined by a set of vertices $\mathcal{V} = \{v_1, \dots, v_R\}$ and a set of connecting edges $\mathcal{E} = \{e_1, \dots, e_D\}$ with $e_d = (v_v, v_u)$, between v_v and its parent v_u . To infer the 3D position of the vertices \mathcal{V} of the tree skeleton graph \mathcal{G} , we use the DDPM formulation proposed by Nunes *et al.* [51] starting with an initialization of the vertex positions obtained by randomly sampling points from the input point cloud. We then predict iterative denoising corrections to distribute the nodes evenly onto the medial axes of the branches. We condition the DDPM predictions with the input point cloud, in order to predict the corresponding tree skeleton out of the learned distribution.

To define the edges \mathcal{E} , for each vertex v_v , we predict the direction of flow of the resin in the tree, which we simply refer to as flow direction, in form of a unit vector pointing towards the parent node v_u . We then generate the full tree skeleton by computing the minimum spanning tree over the nodes based on the predicted node positions and flow directions. We show the process on an exemplary point cloud in Fig. 2. In the following, we first describe the DDPM formulation and then we present our formulation to adapt it to the tree skeletonization task.

3.1. Denoising diffusion probabilistic models

Denoising diffusion probabilistic models [12, 22, 50] sample from the learned distribution by performing an iterative denoising process. The model usually starts from Gaussian noise [12, 22, 50] and at each step removes noise from the input until it converges to the target sample, *e.g.*, images [12, 22, 50, 52, 54, 56, 78, 79] or shapes [44, 45, 58, 59, 69, 72, 76]. This can be achieved by defining a forward diffusion process where noise is iteratively added T times to the target data such that the corrupted data approximates to Gaussian noise. Then, the model is trained to predict the noise that was added at each step t . During inference a novel sample is generated by starting with random noise, and using the trained model to predict and remove the noise iteratively over T steps, arriving to a new sample from the training data.

The diffusion process as formulated by Ho *et al.* [22] can be generally written as follows. Given a sample $\mathbf{x}^0 \sim q(\mathbf{x})$ from a target data distribution q , the diffusion process adds noise to \mathbf{x}^0 over T steps, resulting in $\mathbf{x}^1, \dots, \mathbf{x}^T$, where $q(\mathbf{x}^T) \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathcal{N}(\mathbf{0}, \mathbf{I})$ is a normal distribution with mean $\mathbf{0}$ and the identity matrix \mathbf{I} as diagonal covariance. This diffusion process is parameterized by a sequence of defined noise factors β_1, \dots, β_T , where iteratively at each step t , Gaussian noise is sampled and added to \mathbf{x}^{t-1} given β_t . This can be simplified to sample \mathbf{x}^t from \mathbf{x}^0 , without computing the intermediary steps $\mathbf{x}^1, \dots, \mathbf{x}^{t-1}$. To do so, Ho *et al.* [22] define $\alpha_t = 1 - \beta_t$

and $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$, and \mathbf{x}^t can be sampled as:

$$\mathbf{x}^t = \sqrt{\bar{\alpha}_t} \mathbf{x}^0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, \quad (1)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Note that when T is large enough $q(\mathbf{x}^T) \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$, since $\bar{\alpha}_T$ approaches zero.

For settings where the input data distribution is far from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, Nunes *et al.* [51] propose a point-wise local formulation of the diffusion process. Instead of sampling \mathbf{x}^t as a mixed distribution between $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and \mathbf{x}^0 as in Eq. (1), they formulate the diffusion process as a noise offset added locally to each point $\mathbf{p}_i \in \mathcal{P}$, where \mathcal{P} is the target point cloud to which noise will be added during training. By setting $\mathbf{x}^0 = \mathbf{0}$ in Eq. (1) and adding \mathbf{x}^t to \mathbf{p}_i the diffusion process at timestep t can be formulated as:

$$\mathbf{p}_i^t = \mathbf{p}_i + (\sqrt{\bar{\alpha}_t} \mathbf{0} + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}), \quad (2)$$

$$= \mathbf{p}_i + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, \forall \mathbf{p}_i \in \mathcal{P}. \quad (3)$$

The denoising process aims to undo the T noising steps by predicting the noise $\boldsymbol{\epsilon}$ added at each step t [22]. Given an initial \mathbf{x}^T , we want to reverse the diffusion process and get to \mathbf{x}^0 . The reverse diffusion step can be written as:

$$\mathbf{x}^{t-1} = \mathbf{x}^t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}^t, t) + \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (4)$$

where $\boldsymbol{\epsilon}_\theta(\mathbf{x}^t, t)$ is the noise predicted from \mathbf{x}^t at step t .

This generation can also be guided given a condition \mathbf{c} . This conditional generation can either stem from a pre-trained encoder [12] or from classifier-free guidance [21], where the encoder is trained together with the noise predictor. In our case, we use the classifier-free guidance since it does not require a pre-trained encoder. With the classifier-free guidance, the model is trained to learn the conditional and unconditional noise distribution. In this case, at each training step the model has a probability ρ of predicting the unconditional noise distribution, where the conditioning is set to a null token, *i.e.*, $\mathbf{c} = \emptyset$.

The training process optimizes the denoising model to predict the noise $\boldsymbol{\epsilon}$ added at step t to a given input. Given an input \mathbf{x}^0 and a condition \mathbf{c} , a random step $t \in [0, T]$ is sampled, and \mathbf{x}^t is sampled from Eq. (1) with Gaussian noise $\boldsymbol{\epsilon}$. Then, from \mathbf{x}^t , \mathbf{c} and t , the model computes the noise prediction, supervising it with an \mathcal{L}_2 loss:

$$\mathcal{L}(\mathbf{x}^t, \tilde{\mathbf{c}}, t) = \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \tilde{\mathbf{c}}, t)\|^2, \quad (5)$$

with $\tilde{\mathbf{c}} \sim \mathcal{B}(p)$ where \mathcal{B} is a Bernoulli distribution with outcomes $\{\emptyset, \mathbf{c}\}$ with probability ρ that \emptyset occurs.

The inference starts from an initial $\mathbf{x}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively denoises it to get \mathbf{x}^0 . For the classifier-free guidance [21], we predict the conditional $\boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \mathbf{c}, t)$ and unconditional $\boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \emptyset, t)$ noise distribution and compute the final predicted noise $\boldsymbol{\epsilon}'_\theta(\mathbf{x}^t, \mathbf{c}, t)$ as:

$$\boldsymbol{\epsilon}'_\theta(\mathbf{x}^t, \mathbf{c}, t) = \boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \emptyset, t) + s[\boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \mathbf{c}, t) - \boldsymbol{\epsilon}_\theta(\mathbf{x}^t, \emptyset, t)], \quad (6)$$

where $s \in \mathbb{R}$ is the conditioning weight.

With Eq. (6) we can predict the noise at any step t , from which we can use Eq. (4) to compute $\mathbf{x}^{T-1}, \dots, \mathbf{x}^0$, where \mathbf{x}^0 is a newly generated sample conditioned on c .

3.2. Diffusion tree skeletonization

We want to train the DDPM to predict the skeleton from a point cloud scan of a tree. To do so, we constrain the directed graph to be a rooted tree, therefore each node has a single parent node, as this is true for all topologies of trees stemming from a single growth point. To adapt the formulation of \mathcal{G} to tree skeletonization, we associate each vertex v_r with a 3D position $\mathbf{p}_r \in \mathbb{R}^3$ from the tree skeleton.

Instead of directly predicting the edges \mathcal{E} , we define the set of nodes $\mathcal{M} = \{m_1, \dots, m_D\}$, where each node $m_d = (\mathbf{p}_d, \mathbf{f}_d)$ is composed of its position \mathbf{p}_d and its flow direction \mathbf{f}_d . The flow direction \mathbf{f}_d is a unit vector pointing from the node position \mathbf{p}_d towards the position of its parent node $\mathbf{p}_{\text{par}(m_d)}$. Hence, the flow direction \mathbf{f}_d is defined as:

$$\mathbf{f}_d = \frac{\mathbf{p}_d - \mathbf{p}_{\text{par}(m_d)}}{\|\mathbf{p}_d - \mathbf{p}_{\text{par}(m_d)}\|}. \quad (7)$$

Exemplary nodes and flow directions are shown in Fig. 3.

We then use the DDPM with weights θ to predict a sample $\mathcal{M}_\theta^0 \sim q(\mathcal{M})$. To achieve this we first train our model by adding noise to the nodes m_d as defined in Eq. (3) as:

$$m_d^t = m_d + \sqrt{1 - \bar{\alpha}_t} \epsilon. \quad (8)$$

Training the DDPM to predict the noise ϵ conditioned to an input tree point cloud $\mathcal{S} = \{s_1, \dots, s_B\}$ with $s_b \in \mathbb{R}^3$. Thus, Eq. (5) is written as:

$$\mathcal{L}(\mathcal{M}^t, \mathcal{S}, t) = \|\epsilon - \epsilon_\theta(\mathcal{M}^t, \mathcal{S}, t)\|^2. \quad (9)$$

After the training, we then perform the reverse diffusion step with Eq. (4) given the conditioned noise prediction $\epsilon'_\theta(\mathcal{M}^t, \mathcal{S}, t)$ as in Eq. (6). After iteratively removing the noise over T times, we arrive to a sample $\mathcal{M}_\theta^0 \sim q(\mathcal{M})$ as the skeleton of the input tree point cloud \mathcal{S} from the learned data distribution $q(\mathcal{M})$.

The graph generation process generates a complete graph of the branching structure from the node positions and the flow predictions. We set a subset \mathcal{S}' of the tree point cloud scan \mathcal{S} as the initial corrupted data $\mathcal{M}^T = \mathcal{S}'$, and use the trained DDPM weights θ to denoise \mathcal{M}^T over T times with Eq. (6), arriving to $\mathcal{M}' = \mathcal{M}_\theta^0$. From \mathcal{M}' , we start by defining a graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E})$, with \mathcal{V}' vertices associated with the predicted points \mathbf{p}'_d from the predicted nodes m'_d , and \mathcal{E} being the set of edges between all the predicted nodes. Then, we leverage Kruskal's algorithm [29] to compute the minimum spanning tree $\mathcal{T} = (\mathcal{V}', \mathcal{E}')$ over \mathcal{G}' based on the predicted flow directions \mathbf{f}'_d . To include the

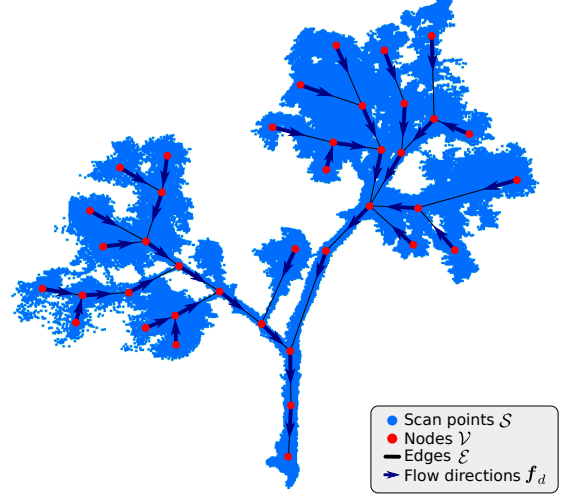


Figure 3. **Tree skeleton graph.** We define the tree skeleton \mathcal{G} as a graph with nodes \mathcal{V} (in red) and edges \mathcal{E} (in black). This graph lies on the medial axis of the branches of the tree given by a point cloud \mathcal{S} (in blue). To generate the graph, our approach predicts the flow direction \mathbf{f}_d (black arrows) at each node position \mathbf{p}_d .

flow directions in the minimum spanning tree computation we define the used edge weights w_{ij} as:

$$w_{ij} = (\mathbf{p}'_i - \mathbf{p}'_j)^\top \mathbf{f}'_i, \quad (10)$$

where \mathbf{p}'_i and \mathbf{p}'_j are the positions of the nodes m'_i and m'_j defining the edge e_{ij} , and \mathbf{f}'_i is the predicted flow direction from the node m'_i .

Kruskal's algorithm then computes the graph that connects all nodes with the minimum total edge weight given the weights computed in Eq. (10), reducing \mathcal{E} to \mathcal{E}' . Thus, the final predicted tree skeleton is defined as $\mathcal{T} = (\mathcal{V}', \mathcal{E}')$, with \mathcal{V}' as the predicted points \mathbf{p}'_d and the edges \mathcal{E}' given the weights computed with the predicted flow direction \mathbf{f}'_d . To further improve the predicted tree topology, we apply a post-processing step where we remove edges that are longer than a threshold l_{\max} and we perform connected components analysis to remove disconnected parts of the graph that contain less than n_{\min} nodes. For more details on the post-processing refer to Sec. B of the supplement.

3.3. Bonn Real-world Tree Skeletonization (BReTS) dataset

To test the performance on real world data, we collected a dataset of 280 apple trees in an orchard. The trees were scanned 10 times over the whole growing season of one year with a terrestrial laser scanner (TLS). As inferring the tree skeleton in the absence of foliage with existing methods proved to give good results, we used the last scan collected after the leaves fell of the trees after the growing phase to

generate the reference skeletons for the other scans. We registered the scans of all rows and all dates in a global reference frame. We then manually cut out the tree rows from the aggregated scans in order to remove ground and support structures. To eliminate the vertical poles in the orchard we performed cylinder fitting on the point cloud and removed all points inside the fitted cylinders. To get a robust initialization for the clustering we removed all points above 0.3 m from the ground and performed HDBSCAN clustering [46] to detect the individual trunk bottoms. To extend the clusters to the rest of the trees, we defined a connectivity graph by defining edges between all points closer than a threshold $d_{max} = 0.2$. We then computed the shortest path distance between all points of the row and the centroid of the initial clusters. To define the cluster membership of all points in the row, we used the shortest path distance in order to obtain the tree instance segmentation. Finally we generated the reference skeletons by running AdTree [15] on the segmented trees and we segmented the point clouds of the other dates by propagating the segmentation of the last date to the other dates by nearest neighbor search. For further details please refer to Sec. A in the supplementary material.

4. Experiments and Discussions

The main focus of this work is an approach that generates a complete tree skeleton covering the wooden parts of a tree from an input point cloud of the tree, including the leaves. To show the capabilities of our approach, we conduct experiments on two different kinds of data: (1) synthetic point clouds and (2) real-world orchard point clouds. The experiments show that our method is able to perform tree skeletonization on different tree species and on both synthetic and real data.

4.1. Experimental setup

Datasets. To test the performance of our method, we evaluate our approach on both synthetic and real-world data.

For the synthetic data, we used the TreeNet3D dataset [68], a multi-variety synthetic dataset that contains point clouds of 13,000 trees of 10 different species generated using the tree modeling software SpeedTree by Orca [61]. As the authors of the dataset did not provide the connectivity information of the nodes but only the connectivity at the branch level, we generated the node hierarchy from the provided branch hierarchy. We also defined the train, validation and test splits by randomly splitting the dataset in a 80%, 10%, and 10% for each tree species, as TreeNet3D does not define them. The simulated scans do not provide color information, so the branch structure needs to be inferred using purely geometric information.

As the synthetic dataset TreeNet3D does not contain apple trees, we performed additional experiments on the simulated apple tree data provided by Dobbs *et al.* [13],

Table 1. **Skeletonization performance on TreeNet3D dataset (normalized by variety)**. Best performance with respect to a particular metric is **bold** and the second best is underlined.

| Approach | Chamfer distance [cm] ↓ | Precision [%] ↑ | Recall [%] ↑ | F1-Score [%] ↑ |
|------------------|-------------------------|-----------------|--------------|----------------|
| AdTree [15] | <u>0.91</u> | 73.64 | 93.72 | <u>79.30</u> |
| LBC [6] | 2.55 | 72.79 | 50.12 | 56.90 |
| PC-Skeletor [48] | 1.37 | 87.20 | 74.37 | 79.00 |
| Smart-Tree [13] | 2.53 | 82.75 | 67.75 | 70.69 |
| DiffTS (Ours) | 0.79 | <u>86.16</u> | <u>83.79</u> | 84.78 |

which contains 100 samples but bridges the gap between TreeNet3D and our orchard dataset, as the simulated point clouds are very dense and provide color information.

To test the performance on real-world data, we evaluated our method on our apple orchard dataset presented in Sec. 3.3, where each tree point cloud has both position and color information with corresponding reference skeletons.

Metrics. To evaluate the tree skeletonization performance, we use commonly used metrics: the Chamfer distance, precision, recall, and F1-score. To this extent, we use uniform distance sampling on the predicted graphs to densely sample a set of points \mathcal{H} from the predicted skeleton and a set of points \mathcal{R} from the reference skeleton. To perform the distance computation between the predicted and the reference skeletons, we compute the Chamfer distance between \mathcal{H} and \mathcal{R} . To further measure the reconstructed skeleton’s quality, we use the F1-score, precision, and recall metrics proposed by Knapitsch *et al.* [28]. We first define precision p and recall r given a threshold δ :

$$p(\delta) = \frac{100}{|\mathcal{R}|} \sum_{\mathbf{r} \in \mathcal{R}} \left[\min_{\mathbf{h} \in \mathcal{H}} \|\mathbf{r} - \mathbf{h}\| < \delta \right], \quad (11)$$

$$r(\delta) = \frac{100}{|\mathcal{H}|} \sum_{\mathbf{h} \in \mathcal{H}} \left[\min_{\mathbf{r} \in \mathcal{R}} \|\mathbf{h} - \mathbf{r}\| < \delta \right], \quad (12)$$

where $\mathbf{h} \in \mathbb{R}^3$ and $\mathbf{r} \in \mathbb{R}^3$ are points from \mathcal{H} and \mathcal{R} , respectively. The operator $[\cdot]$ is the so-called Iverson bracket, *i.e.*, if the condition within the brackets is satisfied, it evaluates to 1, otherwise to 0. We then compute the metrics at 10 thresholds δ in the interval $[\delta_{start}, \delta_{end}]$ and approximate the area under curve. The F1-score $f(\delta)$ is simply the harmonic mean of precision and recall:

$$f(\delta) = \frac{2 \cdot p(\delta) \cdot r(\delta)}{p(\delta) + r(\delta)}. \quad (13)$$

In the tables we report the average of the metrics computed at different thresholds δ . For the evaluation in the Smart-Tree dataset [13], we use the same evaluation parameters proposed by the authors.

Baselines. As baselines, we use state-of-the-art methods for tree skeletonization. Laplacian-based contraction (LBC) [6] iteratively performs contraction of the tree

Table 2. **Skeletonization performance on Smart-Tree apple data branches.** Best performance with respect to a particular metric is **bold** and the second best is underlined.

| Approach | Chamfer distance [cm] ↓ | Precision [%] ↑ | Recall [%] ↑ | F1-Score [%] ↑ |
|-----------------|-------------------------|-----------------|--------------|----------------|
| AdTree [15] | <u>4.55</u> | 29.48 | 49.31 | 36.06 |
| LBC [6] | 8.10 | 23.70 | 7.43 | 11.30 |
| Smart-Tree [13] | 26.55 | 78.72 | 6.53 | 11.84 |
| DiffTS (Ours) | 4.44 | <u>48.07</u> | 28.31 | <u>35.61</u> |

point cloud towards the medial axis of the branches. PC-Skeletor, proposed by Meyer *et al.* [48], improves over LBC by integrating semantic information, the assignment of each point to leaves, branches, or the main trunk, into the skeleton generation process. AdTree [15] creates an initial skeleton by building the minimum spanning tree of the point cloud and then prunes the initial tree skeleton by iteratively removing branches. Smart-Tree [13] uses a neural network to predict the offset vectors pointing from the scan points towards the medial axis of the branches and the offset scan points to build a constrained neighborhood graph that is used with a greedy algorithm to build the tree skeleton.

Implementation details. We based our model on the MinkUnet [10] architecture for the noise prediction necessary for denoising following Eq. (4) and the encoder part of the MinkUnet for computing the conditioning features from the input scan \mathcal{S} . We trained our approach with a learning rate of 10^{-4} using the Adam optimizer [41]. The batch size and the learning rate were tuned on the validation set of the different datasets. As diffusion parameters, we used $T = 1,000$ steps and tuned the noise factors β_1 and β_T for each dataset, as different tree sizes demanded different noise levels. For the full details on the implementation please refer to Sec. C in the supplementary and our open source code.

4.2. Performance on multi-variety synthetic dataset

In the first experiment, we evaluate the performance of our skeletonization approach on the TreeNet3D dataset. This allows us to evaluate the generalization capabilities of our approach to different tree varieties, as this dataset contains trees of 10 different varieties that differ heavily in appearance and scale. To highlight how well our method generalizes, we trained only one model for all 10 tree species. As the different species vary a lot in scale, we normalized the skeletons to have similar canopy sizes before computing the metrics, as otherwise, bigger trees would have a higher weight in the averaged metrics. As seen in Tab. 1, we outperform all baselines in the Chamfer distance and F1-Score, showing that our predicted skeletons are closer to the reference and more accurate. PC-Skeletor [48] is the approach with the best precision, but they use ground truth semantic information as input. AdTree [15] instead performs best

Table 3. **Skeletonization performance on our Orchard dataset.** Best performance with respect to a particular metric is **bold** and the second best is underlined.

| Approach | Chamfer distance [cm] ↓ | Precision [%] ↑ | Recall [%] ↑ | F1-Score [%] ↑ |
|-----------------|-------------------------|-----------------|--------------|----------------|
| AdTree [15] | 4.62 | 14.99 | 54.96 | 22.92 |
| LBC [6] | 6.51 | <u>24.35</u> | 13.46 | 17.19 |
| Smart-Tree [13] | 5.23 | 21.86 | 26.93 | 23.98 |
| TreeQSM [55] | <u>4.60</u> | 17.87 | 48.90 | <u>25.72</u> |
| DiffTS (Ours) | 4.19 | 41.68 | 39.05 | 38.62 |

on the recall metric as the predicted skeletons span over the whole tree nicely, even in tree species with many branches, by using many nodes. However, their precision is lower due to overestimating the existing branches. Due to memory constraints we had to find a tradeoff between efficiency and resolution, which led to predicting 3,000 nodes. Those are way less than the nodes predicted by AdTree, which is a potential limitation. However, this problem leads only to slight degradation in very complex areas that are already error-prone due to occlusions. In fact, our approach still outperforms all baselines in the F1-Score, which gives a more complete picture than precision or recall individually.

4.3. Performance on synthetic apple tree dataset

In the second experiment, we test the performance of our method and the baselines on a synthetic apple tree dataset to bridge the gap between the synthetic multi-species TreeNet3D dataset and our real-world apple orchard dataset. We use the synthetic apple tree data provided by Dobbs *et al.* [13] and use the same evaluation parameters as proposed by the authors. The quantitative results in Tab. 2 show that our method with a value of 7.66 cm achieves the best Chamfer distance compared to the baselines. As shown by the higher precision metric of Smart-Tree [13], that method has the least false positives; however, it has a very low recall value, showing that it misses many branches in the predictions. The small size of this dataset challenges our approach, as one limitation of diffusion models is their reliance on high amounts of training data. Still, it is able to outperform the baselines on the Chamfer distance metric and is a close second on the F1-Score metric, which combines precision and recall.

4.4. Performance on BReTS dataset

In the final experiment, we evaluate the performance of our method on the real-world apple orchard dataset that we presented in Sec. 3.3. This experiment, therefore, tests the real-world applicability of the compared methods, which cannot be shown on synthetic data alone. From the qualitative results in Fig. 4 it immediately becomes clear that the classical approach AdTree [15] does not manage to properly detect where the branches lie. Because most of the points in

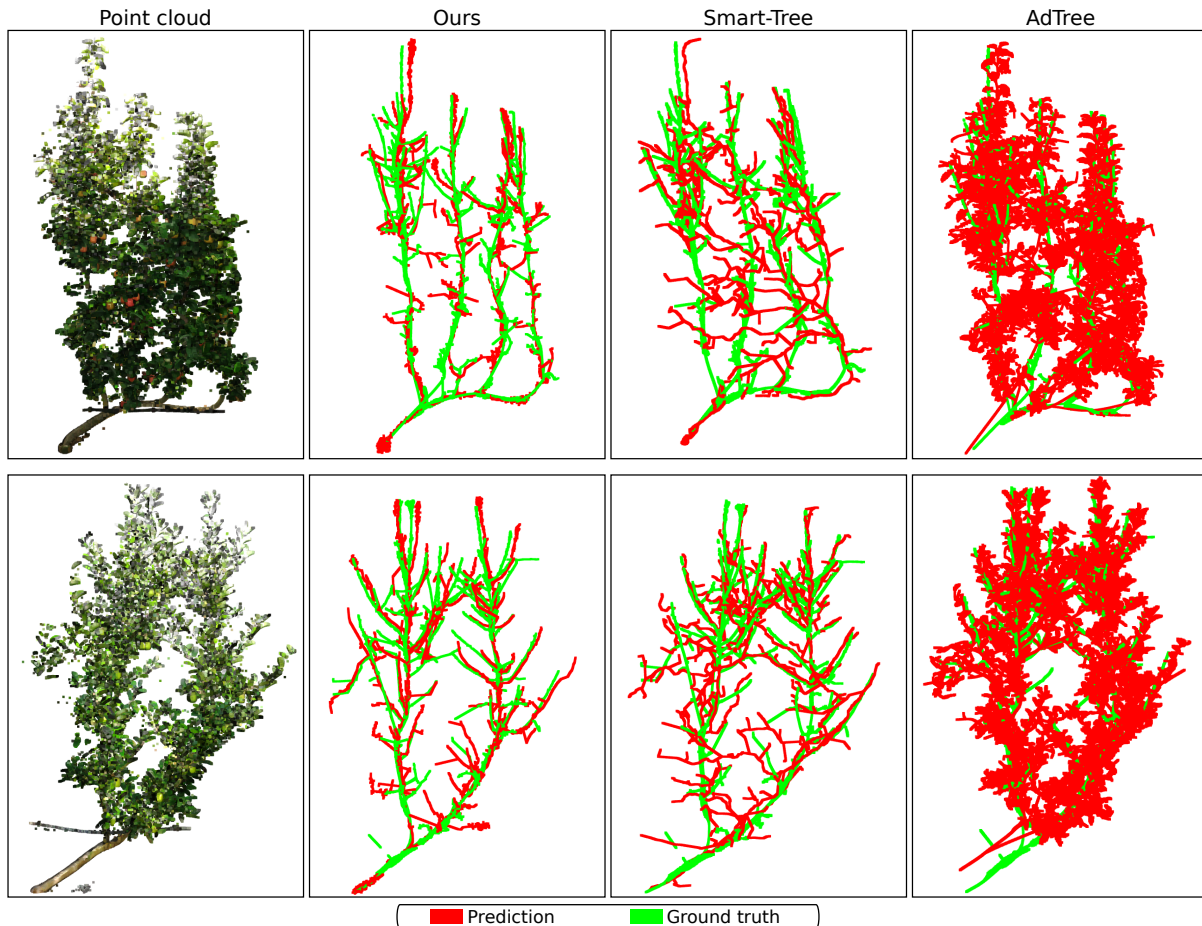


Figure 4. Qualitative results on the BReTS dataset. From the comparison it can be seen that the predictions of our method follow more closely the structure of the reference, and has less additional branches and are more consistent. Notice that our predictions are topological tree structures, however this is not always visible due to perfect overlap with the reference.

the scan are leaves, this is indeed a very hard task without any domain-specific prior knowledge. This domain-specific knowledge is learned during the training process of the models of our approach and Smart-Tree [13], leading to superior identification of the branches. Smart-Tree [13] still has significantly more false positives than our approach, which can also be seen in the precision and Chamfer distance metrics reported in Tab. 3. The quantitative results in Tab. 3 show that our method performs best in all metrics except recall. AdTree [15] obtains the best result in terms of recall, but it becomes clear by visually inspecting the results that this is due to the fact that their predictions have substantially more nodes covering the entire scan. For completeness we also tested the standard deviation of the predictions of our approach with different random initializations which is 0.047 cm on the chamfer distance.

5. Conclusion

In this paper, we proposed a novel tree skeletonization approach that generates a complete tree skeleton covering the

woody parts of the tree from an input point cloud of a tree covered with leaves. We showed that our method is robust to different tree species, scales, and appearances and compared its performance to state-of-the-art methods both on synthetic and real-world data. Our method generally outperforms existing methods on the synthetic datasets, but especially in the real-world setting, the generative nature of the approach has advantages. Due to the extreme amount of occlusions, the learned distribution of tree shapes is very effective. We also propose the first real-world tree dataset with reference skeletons, enabling the quantitative evaluation of tree skeletonization methods on real-world data, which is key to testing their real-world applicability.

Acknowledgments. This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy, EXC-2070 – 390732324 – PhenoRob and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under STA 1051/5-1 within the FOR 5351 (AID4Crops)

References

- [1] Oscar Argudo, Carlos Andújar, and Antoni Chica. Image-Based Tree Variations. *Computer Graphics Forum*, 39(1): 174–184, 2020. 1, 2
- [2] Rajkishan Arikapudi and Stavros Vougioukas. Robotic tree-fruit harvesting with telescoping arms: A study of linear fruit reachability under geometric constraints. *IEEE Access*, 9: 17114–17126, 2021. 1
- [3] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint*, arXiv:2211.01324, 2022. 3
- [4] Jonathan Binney and Gaurav S. Sukhatme. 3d tree reconstruction from laser range data. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2009. 2
- [5] Alexander Bucksch and Roderik Lindenbergh. Campino — a skeletonization method for point cloud processing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63:115–127, 2008. 1, 2
- [6] Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhixun Su. Point Cloud Skeletons via Laplacian Based Contraction. In *Proc. of the Shape Modeling International Conference (SMI)*, pages 187–197, 2010. 1, 6, 7
- [7] Jose L. Cardenas, Carlos J. Ogayar, Francisco R. Feito, and Juan M. Jurado. Modeling of the 3d tree skeleton using real-world data: A survey. *IEEE Trans. on Visualization and Computer Graphics*, 29(12):4920–4935, 2023. 1, 2
- [8] Ayan Chaudhury and Christophe Godin. Skeletonization of Plant Point Cloud Data Using Stochastic Optimization Framework. *Frontiers in Plant Science*, 11, 2020. 1, 2
- [9] Jerome Chave, Maxime Rejou-Mechain, Alberto Burquez, Emmanuel Chidumayo, Matthew S. Colgan, Wellington B.C. Delitti, Alvaro Duque, Tron Eid, Philip M. Fearnside, Rosa C. Goodman, Matieu Henry, Angelina Martínez-Yrizar, Wilson A. Mugasha, Helene C. Muller-Landau, Maurizio Mencuccini, Bruce W. Nelson, Alfred Ngomanda, Euler M. Nogueira, Edgar Ortiz-Malavassi, Raphaël Pélissier, Pierre Ploton, Casey M. Ryan, Juan G. Saldarriaga, and Ghislain Vieilledent. Improved allometric models to estimate the aboveground biomass of tropical trees. *Global Change Biology*, 20(10):3177–3190, 2014. 1
- [10] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7
- [11] Sylvain Delagrangue, Christian Jauvin, and Pascal Rochon. PipeTree: A Tool for Reconstructing Tree Perennial Tissues from Point Clouds. *Sensors*, 14:4271–4289, 2014. 2
- [12] Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat GANs on Image Synthesis. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2021. 3, 4
- [13] Harry Dobbs, Oliver Batchelor, Richard Green, and James Atlas. Smart-tree: Neural medial axis approximation of point clouds for 3d tree skeletonization. In *Proc. of the Iberian Conf. on Pattern Recognition and Image Analysis (ibPRIA)*, 2023. 2, 3, 6, 7, 8
- [14] David Drew, Geoffrey Downes, Thomas Seifert, Annemarie Eckes-Shepard, and Alexis Achim. A review of progress and applications in wood quality modelling. *Current Forestry Reports*, 8:1–16, 2022. 1
- [15] Shenglan Du, Roderik Lindenbergh, Hugo Ledoux, Jantien Stoter, and Liangliang Nan. AdTree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing*, 11(18):2074, 2019. 2, 6, 7, 8
- [16] Simon Ecke, Jan Dempewolf, Julian Frey, Andreas Schwaller, Ewald Endres, Hans-Joachim Klemmt, Dirk Tiede, and Thomas Seifert. UAV-Based Forest Health Monitoring: A Systematic Review. *Remote Sensing*, 14(13):3205, 2022. 1
- [17] Lixian Fu, Ji Liu, Jianlin Zhou, Min Zhang, and Yan Lin. Tree skeletonization for raw point cloud exploiting cylindrical shape prior. *IEEE Access*, 8:27327–27341, 2020. 2
- [18] Linming Gao, Dong Zhang, Nan Li, and Lei Chen. Force field driven skeleton extraction method for point cloud trees. *Earth Science Informatics*, 12:161–171, 2019. 2
- [19] Y. Gong, Y. Yang, and X. Yang. Three-dimensional reconstruction of the virtual plant branching structure based on terrestrial lidar technologies and l-system. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42:403–410, 2018. 2
- [20] Ben Gorte and Norbert Pfeifer. Structuring laser-scanned trees using 3d mathematical morphology. *International Archives of Photogrammetry and Remote Sensing*, 35:929–933, 2004. 2
- [21] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *Proc. of the NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021. 3, 4
- [22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 4
- [23] Harry Huang, Liyu Tang, and Chongcheng Chen. A 3D individual tree modeling technique based on terrestrial LiDAR point cloud data. In *Proc. of the IEEE Intl. Conf. on Spatial Data Mining and Geographical Knowledge Services (ICSDM)*, 2015. 2
- [24] Takahiro Isokane, Fumio Okura, Ayaka Ide, Yasuyuki Matsushita, and Yasushi Yagi. Probabilistic Plant Modeling via Multi-View Image-to-Image Translation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3
- [25] Anling Jiang, Ji Liu, Jianling Zhou, and Min Zhang. Skeleton extraction from point clouds of trees with complex branches via graph contraction. *The Visual Computer*, 37: 1–17, 2021. 2
- [26] Dakai Jin, Krishna Iyer, Cheng Chen, Eric Hoffman, and Punam Saha. A Robust and Efficient Curve Skeletonization Algorithm for Tree-Like Objects Using Minimum Cost Paths. *Pattern Recognition Letters*, 76:32–40, 2015. 1, 2
- [27] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 3

- [28] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. on Graphics*, 36(4):1–13, 2017. 6
- [29] Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. of the American Mathematical Society*, 7(1):48–50, 1956. 5
- [30] Jumin Lee, Woobin Im, Sebin Lee, and Sung-Eui Yoon. Diffusion probabilistic models for scene-scale 3d categorical data. *arXiv preprint*, arXiv:2301.00527, 2023. 3
- [31] Jae Lee, Bosheng Li, and Bedrich Benes. Latent l-systems: Transformer-based tree generator. *ACM Trans. on Graphics (TOG)*, 43(1):1–16, 2023. 2
- [32] Bosheng Li, Jacek Kałuzny, Jonathan Klein, Dominik Michels, Wojtek Palubicki, Bedrich Benes, and Sören Pirk. Learning to reconstruct botanical trees from single images. *ACM Trans. on Graphics (TOG)*, 40:1–15, 2021. 1, 2
- [33] Yuan Li, Zhihao Liu, Bedrich Benes, Xiaopeng Zhang, and Jianwei Guo. SVDTree: Semantic Voxel Diffusion for Single Image Tree Reconstruction. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3
- [34] Cheng Lin, Changjian Li, Yuan Liu, Nenglu Chen, Yi-King Choi, and Wenping Wang. Point2Skeleton: Learning Skeletal Representations from Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2
- [35] Aristid Lindenmayer. Mathematical models for cellular interactions in development ii. simple and branching filaments with two-sided inputs. *Journal of Theoretical Biology*, 18(3):300–315, 1968. 2
- [36] Xinpeng Liu, Hiroaki Santo, Yosuke Toda, and Fumio Okura. TreeFormer: Single-view plant skeleton estimation via tree-constrained graph generation. In *Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2025. 3
- [37] Yanchao Liu, Jianwei Guo, Bedrich Benes, Oliver Deussen, Xiaopeng Zhang, and Hui Huang. TreePartNet: neural decomposition of point clouds for 3D tree reconstruction. *ACM Trans. on Graphics (TOG)*, 40:1–16, 2021. 2
- [38] Zhihao Liu, Kai Wu, Jianwei Guo, Yunhai Wang, Oliver Deussen, and Zhanglin Cheng. Single image tree reconstruction via adversarial network. *Graphical Models*, 117:101115, 2021. 1, 2
- [39] Zhihao Liu, Yu Li, Fangyuan Tu, Ruiyuan Zhang, Zhanglin Cheng, and Naoto Yokoya. Deeptreesketch: Neural graph prediction for faithful 3d tree modeling from sketches. In *Proc. of the CHI Conf. on Human Factors in Computing Systems*, 2024. 3
- [40] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang, and Jihad El-Sana. Automatic Reconstruction of Tree Skeletal Structures from Point Clouds. *ACM Trans. on Graphics (TOG)*, 29:151, 2010. 2
- [41] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2019. 7
- [42] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 3
- [43] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22(4):730–751, 2025. 3
- [44] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 4
- [45] Zhaoyang Lyu, Zhifeng Kong, Xudong XU, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2022. 3, 4
- [46] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205, 2017. 6
- [47] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3
- [48] Lukas Meyer, Andreas Gilson, Oliver Scholz, and Marc Stamminger. CherryPicker: Semantic Skeletonization and Topological Reconstruction of Cherry Trees. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, 2023. 3, 6, 7
- [49] Kazuto Nakashima and Ryo Kurazume. LiDAR Data Synthesis with Denoising Diffusion Probabilistic Models. *arXiv preprint*, arXiv:2309.09256, 2023. 3
- [50] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, 2021. 3, 4
- [51] Lucas Nunes, Rodrigo Marcuzzi, Benedikt Mersch, Jens Behley, and Cyrill Stachniss. Scaling Diffusion Models to Real-World 3D LiDAR Scene Completion. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 4
- [52] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023. 3, 4
- [53] Chakkrit Preuksakarn, Frédéric Boudon, Pascal Ferraro, Jean-Baptiste Durand, Ekko Nikinmaa, and Christophe Godin. Reconstructing Plant Architecture from 3D Laser scanner data. In *Proc. of the 6th International Workshop on Functional-Structural Plant Models*, 2010. 2
- [54] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, 2021. 3, 4
- [55] Pasi Raunonen, Mikko Kaasalainen, Markku Åkerblom, Sanna Kaasalainen, Harri Kaartinen, Mikko Vastaranta, Markus Holopainen, Mathias Disney, and Philip Lewis. Fast

- automatic precision tree models from terrestrial laser scanner data. *Remote Sensing*, 5(2):491–520, 2013. 7
- [56] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 4
- [57] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2022. 3
- [58] Aditya Sanghi, Hang Chu, Joseph G. Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshah. CLIP-Forge: Towards Zero-Shot Text-To-Shape Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 4
- [59] Aditya Sanghi, Rao Fu, Vivian Liu, Karl D.D. Willis, Hooman Shayani, Amir H. Khasahmadi, Srinath Sridhar, and Daniel Ritchie. CLIP-Sculptor: Zero-Shot Generation of High-Fidelity and Diverse Shapes From Natural Language. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3, 4
- [60] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2021. 3
- [61] SpeedTree. Speedtree, open research content archive (orca), 2017. <http://developer.nvidia.com/orca/speedtree>. 6
- [62] Ondrej Stava, Sören Pirk, Julian Kratt, Baoquan Chen, Radomír Měch, Oliver Deussen, and Bedrich Benes. Inverse procedural modelling of trees. *Comput. Graph. Forum*, 33(6):118–131, 2014. 2
- [63] Zhixun Su, Zhao Yuandi, Chunjiang Zhao, Xinyu Guo, and Zhiyang Li. Skeleton extraction for tree models. *Mathematical and Computer Modelling*, 54:1115–1120, 2011. 2
- [64] Zhonghua Su, Shihua Li, Hanhu Liu, and Ze He. Tree skeleton extraction from laser scanned points. In *Proc. of the IEEE Intl. Geoscience and Remote Sensing Symposium (IGARSS)*, 2019. 2
- [65] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM Trans. on Graphics (TOG)*, 28(3):71, 2009. 1, 2
- [66] Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 3d skeletons: A state-of-the-art report. *Computer Graphics Forum*, 35(2):573–597, 2016. 2
- [67] Lina Tang and Guofan Shao. Drone remote sensing for forestry research and practices. *Journal of Forestry Research*, 26:791–797, 2015. 1
- [68] Shengjun Tang, Zhuoyu Ao, Yaoyu Li, Hongsheng Huang, Linfu Xie, Ruisheng Wang, Weixi Wang, and Renzhong Guo. Treenet3d: A large scale tree benchmark for 3d tree modeling, carbon storage estimation and tree segmentation. *Intl. Journal of Applied Earth Observation and Geoinformation*, 130:103903, 2024. 3, 6
- [69] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3D: Zero-Shot Text-to-3D Synthesis Using 3D Shape Prior and Text-to-Image Diffusion Models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3, 4
- [70] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. *arXiv preprint*, arXiv:2012.03762, 2020. 3
- [71] Azlan Zahid, Md Sultan Mahmud, Long He, Paul Heineemann, Daeun Choi, and James Schupp. Technological advancements towards developing a robotic pruner for apple trees: A review. *Computers and Electronics in Agriculture*, 189:106383, 2021. 1
- [72] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3, 4
- [73] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023. 3
- [74] Xiaopeng Zhang, Hongjun Li, Mingrui Dai, Wei ma, and Long Quan. Data-Driven Synthetic Modeling of Trees. *IEEE Trans. on Visualization and Computer Graphics*, 20:1214–1226, 2014. 2
- [75] Wang Zhen, Liqiang Zhang, Tian Fang, P. Takis Mathiopoulos, Huamin Qu, Chen Dong, and Wang Yuebin. A Structure-Aware Global Optimization Method for Reconstructing 3-D Tree Models From Terrestrial Laser Scanning Data. *IEEE Trans. on Geoscience and Remote Sensing*, 52(9):5653–5669, 2014. 2
- [76] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D Shape Generation and Completion Through Point-Voxel Diffusion. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 3, 4
- [77] Xiaochen Zhou, Bosheng Li, Bedrich Benes, Songlin Fei, and Soren Pirk. Deeptree: Modeling trees with situated latents. *IEEE Trans. on Visualization and Computer Graphics*, 30(8):5795–5809, 2023. 2
- [78] Yufan Zhou, Ruiyi Zhang, Changyou Chen, Chunyuan Li, Chris Tensmeyer, Tong Yu, Jiuxiang Gu, Jinhui Xu, and Tong Sun. Towards Language-Free Training for Text-to-Image Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 4
- [79] Yufan Zhou, Bingchen Liu, Yizhe Zhu, Xiao Yang, Changyou Chen, and Jinhui Xu. Shifted Diffusion for Text-to-Image Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3, 4
- [80] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to Generate Realistic LiDAR Point Clouds. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022. 3