Supplementary Material
Tree Skeletonization from 3D Point Clouds by Denoising Diffusion

In this supplementary material, we provide in Sec. A additional details about how we collected and processed the dataset introduced in Sec. 3.3, in Sec. B about how the post-processing of our approach works, and further experimental results in Sec. C.

A. Our Dataset

We recorded our dataset using the terrestrial laser scanner (TLS) Focus3D X 130 by FARO that we manually placed at multiple locations along the orchard rows. The resulting point clouds have an average nearest-neighbor distance of 2.7 mm on the relevant parts that are covering the trees. We recorded data in 3 driving lanes between the trees and for each of the lanes, we recorded 10 scans with the TLS at a distance of around 9 m from each other. This leads to a total of 30 scans comprising 4 rows of apple trees recorded from at least one side. As our goal is to estimate the skeleton structure of trees when the foliage is present, we collected data in the same orchards at different points in time, over a span of 8 months. Specifically, we recorded data in early spring when only the woody parts are present (see Fig. 1 (a)), in later spring when flowers are on the trees (see Fig. 1 (b)), in early summer when the canopies are still developing (see Fig. 2 (a)) and in late summer when the canopies are fully developed (see Fig. 2 (b)). The dataset was recorded at the orchard facilities of the University of Bonn at Campus Klein-Altendorf, Germany.

A.1. Intra-date Scan Registration

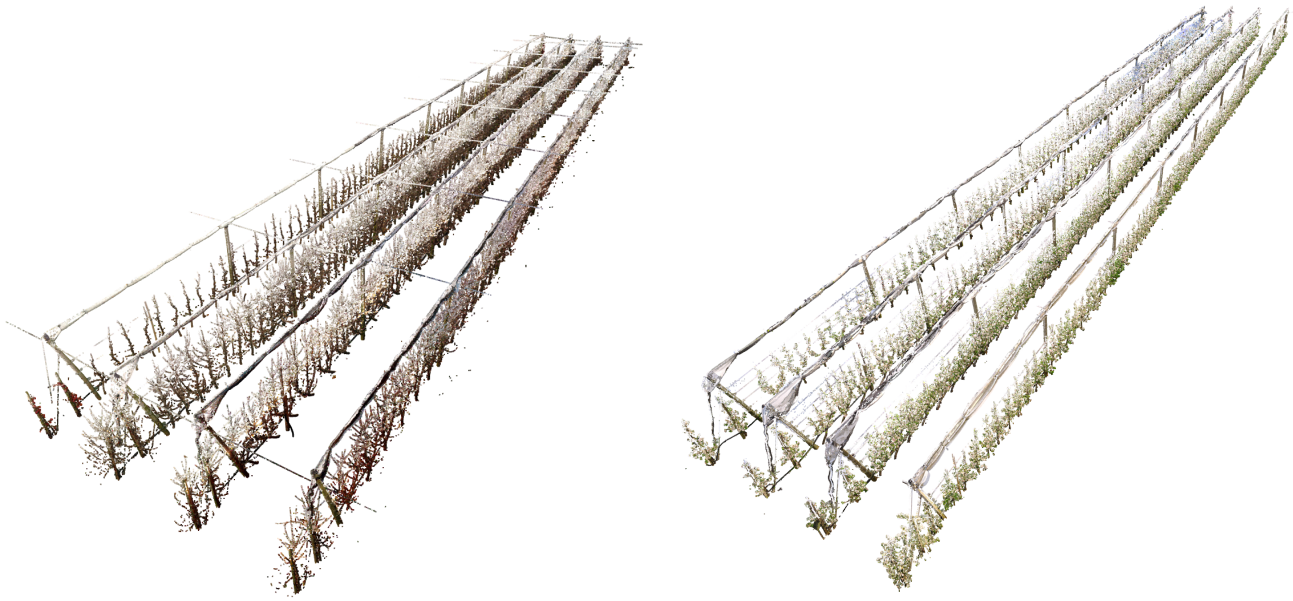
Given that we used a TLS to collect our dataset, after each data collection campaign, we end up with 30 individual scans. To aggregate each scan into a single point cloud we use a multi-step registration approach. First we find scan-to-scan transformations for neighboring scans, both in the same row and in the neighboring rows, leading to each scan having 4 neighbors. We compute these transformations using RANSAC [5] to obtain an initial guess and Iterative Closest Point (ICP) to perform a refinement of the transformations. Afterwards, we define a factor graph where the nodes are the poses of the scans, and the transformations found in the previous step represent the factors. To reject inaccurate transformations, we perform the factor graph optimization multiple times, iteratively dropping the factors with the biggest residuals and optimizing again. By doing this, we drop the 4 transformations that are the least coherent with all others, leading to a better final estimation of the global pose of each individual scan. While this could potentially lead to disconnected nodes in the graph, it did not happen in practice. To perform further refinement after the scan-to-scan registration, we perform multi-scan registration, where each scan gets matched to every other scan in one big optimization process, as proposed by Wiesmann et al. [9]. This leads to well-aligned aggregated point cloud, where also points far from the local origin of each scan are aligned well to the rest of the points.

A.2. Inter-date Scan Registration

To globally align each scan over the growing season, we also performed registration of the scans collected at different points in time. For this purpose, we selected the date with the best intra-date scan registration, indicating the best global consistency, based on visual inspection, and defined it as reference date. Then, to align the other dates, we perform again RANSAC and ICP, however due to the big variation in the canopy appearance caused by different amounts of foliage, we used only the lower trunks for the initial guess estimation. To ensure proper alignment of all parts of the covered orchard, we then perform again multi-scan registration as proposed by Wiesmann et al. [9], using the reference map as an additional scan in the optimization.

A.3. Instance Segmentation

After registering the aggregated point clouds from different dates, we segment individual trees. As segmenting trees is easiest when no leaves are present we run an ad-hoc unsupervised clustering approach on the last date, where shoots have fully developed and leaves have completely fallen. As preprocessing step, we aim to remove all points that are not part of the trees. To do so, we use RANSAC plane fitting to segment out all points that lie within 10 cm from the fitted ground plane. Then, we manually click the top of the poles that are present in the rows and perform cylinder fitting using RANSAC and ICP to remove all points that lay in the 2 cm neighborhood of the found cylinders. To get the



(a) (b)

Figure 1. An overview of the orchard without leaves in (a), and with flowers in (b). The white color on top of the trees is caused by the sensor used during data recording.

initial clusters T_{init}, we apply HDBSCAN clustering on the points right above the ground, i.e., all points with a positive distance between 10 cm and 30 cm from the ground plane which we found through RANSAC on the whole point cloud. These points represent the trunks, which are structures that are stable throughout the whole season and easy to cluster, as they are spatially non overlapping. We then compute the KNN-neighbor-graph with $K = 50$, over the whole point cloud, and assign each point to the cluster in T_{init} that is closest on the graph. By using the cube of the euclidean distance as the graph distance metric, we ensure that edges along the branches have low distance, while gaps between branches are very distant, leading to very good tree instance segmentation results.

Once the tree instances are defined on the last date, we can obtain instances in the other dates by propagating the labels through nearest neighbor search, since the point clouds are registered in a common reference frame.

A.4. Reference Skeleton Computation

To compute the reference skeletons for our dataset we leveraged the good performance of AdTree on extracting tree skeletons from tree data without foliage. We computed the skeletons using the segmented trees on the last data collection date, where the branches are fully developed and the foliage has fallen completely due to the cold season. As the point clouds of all dates are in the same reference frame, the extracted skeletons can than be used as reference for the other dates, where the extraction presents many more challenges due to the high level of occlusion by leaves and fruits. To account for the growth of the branches during the growing season, we adapt the reference skeleton to each point cloud by removing all nodes and edges from the reference skeleton graph that are further than 20 cm from the point cloud itself.

B. Postprocessing of Tree Skeletons

To improve the skeletons resulting from the skeletonization process, we perform postprocessing on the predicted skeletons. Given that we do not want to reconstruct branches that are further than 20 cm away from the closest point in the input point cloud, we compute the distance of each node in the graph to the input point cloud and discard the nodes which have a distance to the closest point greater than 20 cm. Furthermore, we noticed that while our approach leads to skeletons that are densely covering the branching structure with nodes, sometimes nodes are wrongly connected between near branches. Given this high density of nodes predicted on the branching structure, the few existing long edges are mostly these wrong connections between different branches. Therefore, in the post processing we eliminate edges longer than a threshold of 4 cm which empirically gave the best result on the validation set. After these long edges have been removed, we cluster the remaining graph with connected component and discard components that have fewer than 20 nodes. Then to reconnect all parts of the graph into one tree again, we perform Minimum Spanning Tree search and obtain the final tree skeleton.

(a) (b)

Figure 2. An overview of the orchard without little leaves in (a), and with many leaves in (b). The white color on top of the trees is caused by the sensor used during data recording.

C. Architecture

Our tree skeletonization approach uses the architecture proposed by Nunes et al. [7] with small modifications. To predict the noise at each step t, we used a MinkUnet [2] to predict the noise for each point. For the condition encoder, we used only the encoder part of the MinkUNet with the same architecture as the noise predictor. Instead of taking only the closest point in the conditioning cloud $\mathcal{C}$ to condition each node in $\mathcal{G}$ we take K nearest neighbors, with $K = 8$. We then compute the conditioning feature by computing the weighted mean over the K neighboring features, with an inverse distance weighting. As described in Sec. 3.1 of the main paper, before each layer l, we compute the positional embeddings τ from the denoising step t with an embedding dimension $d_t = 96$, conditioning the layer input $\mathcal{F}_l$ to $\mathcal{C}$ and t with the conditioning block. Fig. 4 shows the architecture of the conditioning encoder and the noise predictor, with each layer l features dimension d_l and the conditioning scheme. The above architectures with the specified feature sizes result in 8.5 M parameters for the condition encoder model and 26.8 M parameters for the noise predictor model.

D. TreeNet3D Variety-Wise Results

To avoid larger varieties dominating the metrics due to the large-scale variance in TreeNet3D (8-126 m, from Fig. 2 in [8]), we normalized by variety before computing the values reported in Tab. 1 of the main paper. To give a more detailed presentation of the results and show the unnormalized errors we report the Chamfer distance for each variety individually in Tab. 1.

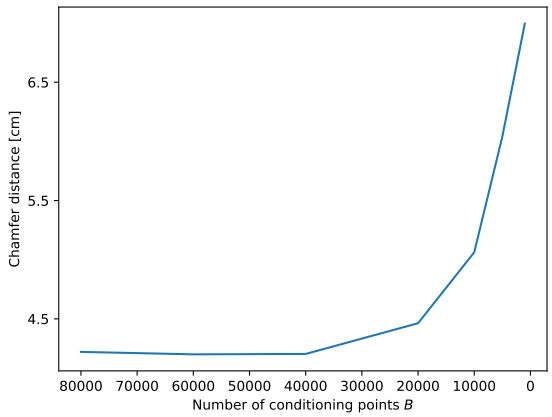


Figure 3. Evolving of Skeletonization performance measured by chamfer distance with varying input scan densities.

E. Ablation on Number of Conditioning Points

To test the robustness of our approach to different amounts of points B in the input scans $\mathcal{S}$, we performed an ablation study by reducing the input points $\mathcal{S}$ more and more, showing how the performance decreases on the orchard dataset. We show the results in the plot in Fig.3.

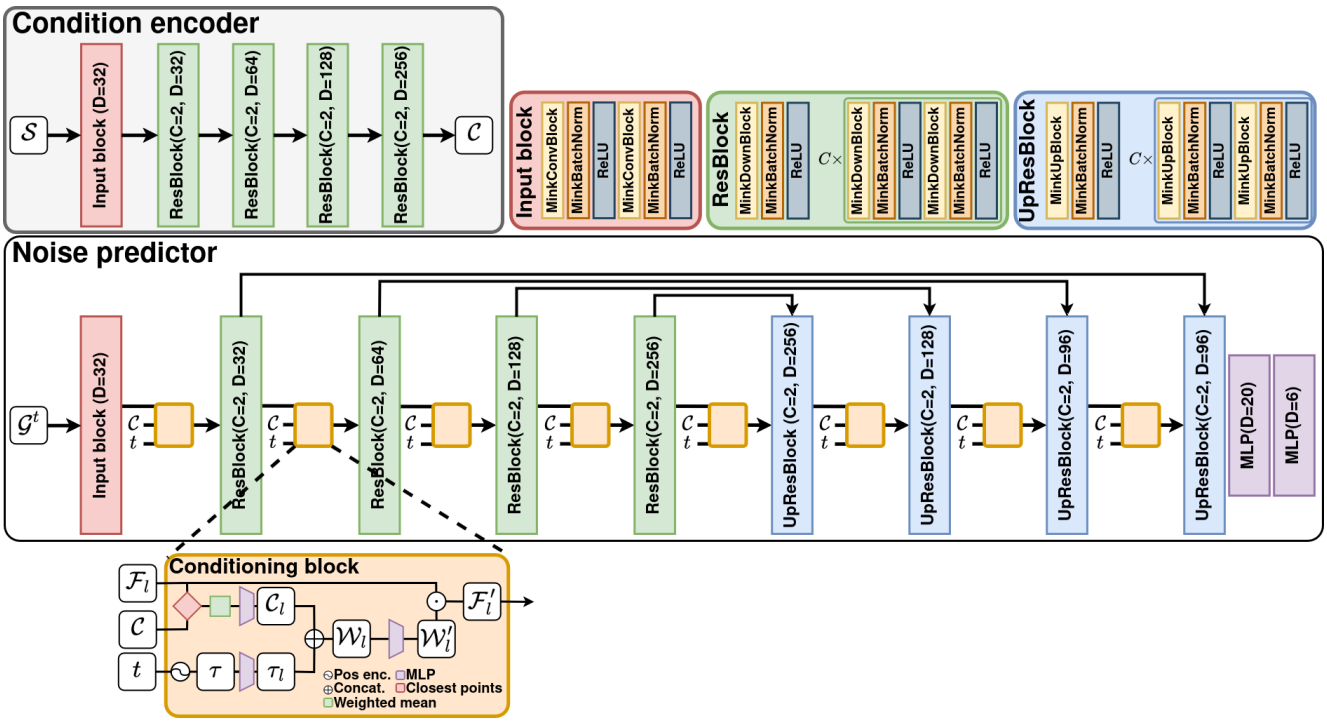


Figure 4. Noise predictor and condition encoder models architecture. The condition encoder receives the scan $\mathcal{S}$ and computes the conditioning point cloud $\mathcal{C}$. From t, we compute the positional embedding τ with a dimension $d_t = 96$. At each layer l, we give $\mathcal{C}$ and τ to the conditioning block together with the layer input features $\mathcal{F}_l$ to get $\mathcal{F}'_l$, which is then feed as input to the layer l.

Table 1. **Skeletonization chamfer distance without normalization on TreeNet3D dataset by variety.** Some varieties are abbreviated for space reasons, the full names being Agarwood, Ajianglanren, FloodedGum, Lemon, Lombardy Poplar, Shanshu, Tibetan Cherry, Xiaoyelanren, Zhangshu. Best performance is **bold** and the second best is underlined.

Approach	Agarwood	Ajiangl	FloodedGum	Lemon	Poplar	Shanshu	Cherry	Xiaoyel	Zhangshu
AdTree [4]	**10.89**	49.50	15.90	**3.85**	**4.45**	18.36	**5.03**	**2.03**	**7.16**
LBC [1]	30.40	139.58	70.55	23.34	92.45	325.49	131.27	100.31	84.54
PC-Skeletor [6]	30.66	57.36	27.57	35.30	76.41	211.24	135.14	659.76	119.93
Smart-Tree [3]	26.55	101.85	27.45	10.65	9.83	17.11	10.27	69.30	21.41
Ours	15.99	**14.03**	**15.50**	6.42	8.30	**5.06**	15.18	3.37	21.64

References

[1] 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. 4

[2] 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. 3

[3] 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. 4

[4] 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. 4

[5] M.A. Fischler and R.C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981. 1

[6] 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. 4

[7] 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

[8] 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

[9] L. Wiesmann, E. Marks, S. Gupta, T. Guadagnino, J. Behley, and C. Stachniss. Efficient LiDAR Bundle Adjustment for Multi-Scan Alignment Utilizing Continuous-Time Trajectories. *arXiv preprint*, arXiv:2412.11760, 2024. 1