TreeFlow: Conditional Flow Matching for 3D Tree Point Cloud Generation from Inventory Attributes

Anthony Marcozzi *, Johnathan Tenny, Daithi Martin, Juan Castorena, Zachary Crennen, Lucas Wells, Samuel Hillman

Posted Date: 27 April 2026

doi: 10.20944/preprints202604.1825.v1

Keywords: flow matching; point cloud generation; 3D tree structure; terrestrial laser scanning; conditional generative modeling; crown architecture; forest digital twin; deep learning; wildland fuel; fuel modeling

 doi:10.20944/preprints202604.1825.v1

Article

TreeFlow: Conditional Flow Matching for 3D Tree Point Cloud Generation from Inventory Attributes

Anthony Marcozzi [1,*], Johnathan Tenny [2], Daithi Martin [1], Juan Castorena [3], Zachary Crennen [3], Lucas Wells [4] and Samuel Hillman [5]

[1] Center for Applied Fire and Ecosystem Science, New Mexico Consortium, Los Alamos, NM, USA
[2] School of Informatics, Computing, and Cyber Systems, Northern Arizona University, Flagstaff, AZ, USA
[3] Los Alamos National Laboratory, Los Alamos, New Mexico, USA
[4] Silvx Labs, Missoula, MT, USA
[5] Department of Energy, Environment and Climate Action, Government of Victoria, East Melbourne, Victoria, Australia
* Correspondence: amarcozzi@newmexicoconsortium.org

Abstract

Accurate three-dimensional representations of tree structure are essential for fire modeling, radiative transfer simulation, synthetic data generation, and digital twins of forests, yet detailed 3D structure is rarely available at required scales. Current approaches approximate crowns with smooth geometric primitives, discarding the clumping, gaps, and irregular branching present in real trees. We present TreeFlow, a conditional flow matching model that generates realistic 3D tree point clouds from species, acquisition platform, and height. The model uses a transformer trained on real laser scanning data from the FOR-species20K benchmark to learn a velocity field transporting samples from a Gaussian distribution to the source data distribution. We evaluate generation quality by comparing conditioning and distributional fidelity metrics to scans of real trees. Generated trees match or approach the intra-class baseline on five of six metrics, with a Chamfer distance of 0.581 m versus 0.559 m for real trees of the same genus and height class. Performance is strongest below 25 m and degrades with increasing height. TreeFlow is the first flow matching model to produce 3D tree point clouds from scalar inventory attributes using real laser scanning data.

Keywords: flow matching; point cloud generation; 3D tree structure; terrestrial laser scanning; conditional generative modeling; crown architecture; forest digital twin; deep learning; wildland fuel; fuel modeling

1. Introduction

Three-dimensional vegetation structure plays a central role in forest ecology and management. The sizes, shapes, and spatial arrangements of individual tree crowns determine canopy light interception, influence competition and growth dynamics, shape the turbulent wind flow environment within and above forest canopies, and control the distribution and dynamics of wildland fuels. Accurate 3D representations of trees are needed to generate high-resolution fuel inputs for next-generation fire models [1,2], simulate radiative transfer through forest canopies [3–5], create synthetic training data for remote sensing algorithms [6,7], and construct digital twins of forests [8]. In each of these applications, the accuracy of downstream predictions is sensitive to the fidelity of 3D vegetation inputs.

Despite its importance, detailed 3D tree structure is rarely available at the scales these applications require. Most operational systems approximate crown shape using approximate geometric forms, commonly assuming uniform fuel distribution within crowns [5,9,10]. FastFuels constructs individual tree crowns using crown profile models within parametric envelopes [11,12]. Forest growth simulators represent crowns as ellipsoids, paraboloids, or cylinders scaled by allometric relationships [13]. These functional forms are smooth and symmetric by construction and cannot represent the clumping, gaps, or differential branching patterns present in real crowns. Vegetation structure influences fire behavior

 doi:10.20944/preprints202604.1825.v1

not only through its distribution of combustible mass but also through its effects on fluid drag, wind entrainment, and microclimate [14–17].

Advances in remote sensing have made it increasingly possible to measure individual trees across large areas. Airborne laser scanning (ALS) and uncrewed aerial system (UAS) photogrammetry can detect individual trees and estimate their height, crown dimensions, and in some cases diameter, though detection is biased toward dominant overstory trees, and results can be described as tree-approximate objects rather than true individual tree measurements [18]. Structure-from-motion (SfM) methods applied to UAS imagery have demonstrated the ability to extract tree locations, heights, and diameter with low error in open-canopy conifer forests [19,20]. High-resolution canopy height models derived from aerial imagery now cover the entire conterminous United States at sub-meter resolution [21]. Terrestrial laser scanning (TLS) and ground-based mobile laser scanning (MLS) captures detailed 3D structure at plot scales, and emerging operational protocols such as IntELiMon are expanding the footprint of TLS acquisitions for ecosystem and fire effects monitoring [22]. Combining these data sources with predictive models for unmeasured attributes creates the possibility of generating census-like tree inventories [23]. However, existing approaches exhibit a fundamental trade-off between spatial coverage and structural detail. TLS captures fine-scale internal crown architecture but is limited to plot-scale measurements. Conversely, methods that support landscape-scale inventories, such as ALS, UAS SfM, and canopy height models derived from active and passive satellite sensors primarily describe the outer canopy surface, producing tree-level attributes (e.g. position, height, and crown width) without resolving within-crown structure.

Bridging this gap has traditionally required chaining together multiple submodels. A typical pipeline uses allometric equations to predict crown dimensions from diameter and height, a crown profile model to define the outer envelope, and a probabilistic distribution function to populate the interior with fuel mass [1,2]. Each submodel contains significant assumptions, many of them untested, and when linked together errors propagate and compound in canopy structure models and downstream model outputs.

An alternative is to bypass parametric models by using tree data clipped directly from TLS point clouds. Schäfer et al. [24] matched inventory trees to a database of real scanned trees by species and dimensions, then assembled and virtually scanned them with HELIOS++. Li et al. [25] transformed TLS scans of reference trees to match ALS-detected tree positions and dimensions across larger areas. Both approaches avoid geometric simplifications but depend on the availability of TLS-scanned trees that closely match the target specifications. Existing scan collections cover limited geographic regions and species, and insufficient data could lead to excessive scaling or reuse of some tree entities, leading to undesirable results.

Large-scale aggregation of very high resolution point cloud datasets could enable new opportunities for simulating forest architecture using individually scanned trees. The FOR-species20K benchmark dataset [26] comprises 20,158 point clouds representing manually segmented individual trees spanning 33 species from 19 genera, collected primarily from European forests using TLS, MLS, and ULS platforms. The dataset covers temperate, boreal, and Mediterranean biogeographic regions with tree heights ranging from 0.3 m to 56.3 m. Originally developed for benchmarking species classification models, its scale and diversity make it feasible to train generative models that learn 3D tree structure directly from real data. Recent advances in generative modeling, particularly diffusion models and flow matching [27,28], have demonstrated the ability to learn complex distributions over 3D point clouds and generate realistic samples [29,30].

Several efforts have applied these and related methods to tree modeling. At the individual tree level, Lee et al. [31] trained a transformer to generate L-system strings from procedurally generated trees, Zhou et al. [32] learned branch development rules through a situated latent space, Wang et al. [33] developed an autoregressive transformer for branch skeleton generation, Lee et al. [34] generated 600,000 3D tree models from single images using diffusion priors, and Xu et al. [35] applied diffusion to tree point cloud generation but trained on small synthetic datasets without conditioning on tree

 doi:10.20944/preprints202604.1825.v1

attributes. All of these methods are trained on procedurally generated or synthetic trees rather than real laser scanning data. Stone et al. [36] and Bornand et al. [37] explored the use of synthetic tree data for training downstream deep learning tasks and found that the realism of synthetic training data significantly affects performance on real data. At larger scales, Castorena et al. [38] and Hartsook et al. [39] used real LiDAR data to generate TLS-resolution forest structure from ALS inputs. No existing method generates 3D individual tree point clouds from scalar inventory attributes using a model trained on real laser scanning data.

In this paper, we present TreeFlow, a conditional flow matching model that generates 3D individual tree point clouds from user-input species, acquisition platform, and tree height. The model learns to transform samples from a standard Gaussian distribution into realistic tree point clouds by integrating a learned velocity field conditioned on tree attributes. We train on the FOR-species20K dataset [26] using a transformer architecture with long skip connections. We evaluate generation quality along two axes: conditioning fidelity, which measures whether individual generated trees reproduce the geometric properties of their source trees, and distributional fidelity, which measures whether the population of generated trees reproduces the statistical structure of the real population. To our knowledge, this represents the first application of generative flow matching to produce 3D tree point clouds from scalar inventory attributes using a model trained entirely on real laser scanning data.

2. Methods

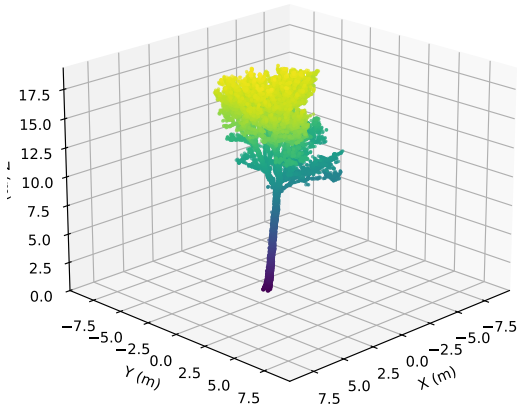
2.1. Data

We trained our generative model using the FOR-species20K benchmark dataset, a large-scale collection of individual tree point clouds originally developed for species classification from proximally sensed laser scanning data [26]. The full dataset comprises 20,158 manually segmented trees spanning 33 species from 19 genera, with a minimum of 50 individuals per species. Of these, 17,707 are released with species labels as the development set. The remaining 2,451 have withheld labels for the classification benchmark and are not used in this study. For each tree, the dataset provides a 3D point cloud along with metadata including species, acquisition platform, and tree height (Figure 1).

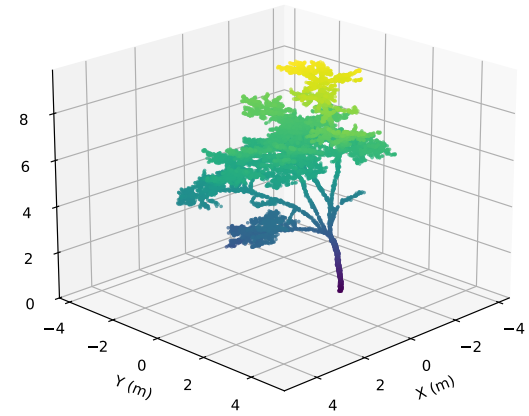
Data represent temperate (61%), boreal (25%), and Mediterranean (7%) biogeographic regions, with collections occurring mainly in European forests with some additional samples from Canada, Australia, and New Zealand. The dataset incorporates point clouds from three acquisition platforms: terrestrial laser scanning (TLS, 70%), UAS laser scanning (ULS, 22%), and mobile laser scanning (MLS, 8%), captured using more than 12 different sensors. This multi-platform composition introduces substantial variation in point density, occlusion patterns, and spatial resolution. Tree heights range from 0.3 m to 56.3 m, encompassing developmental stages from samplings to mature trees.

Point clouds were preprocessed through a three-step normalization procedure designed to facilitate stable flow matching training. First, each point cloud was translated to a common origin defined as the center of its 3D bounding box, aligning the target distribution with the zero-centered Gaussian source distribution. Second, coordinates were scaled by tree height, normalizing the vertical extent of all trees to a unit interval of $[-0.5, 0.5]$ while preserving natural aspect ratios. Third, an additional scaling factor of 2.0 was applied, stretching the vertical extent to $[-1, 1]$ to match the unit standard deviation of the Gaussian source distribution. This normalization helps ensure that the learned velocity field does not need to account for large translations or scale differences, improving training stability and enabling efficient integration during sampling. Lastly, the dataset was divided into training (80%), validation (10%), and test (10%) subsets using stratified sampling by species to ensure representation across all splits.

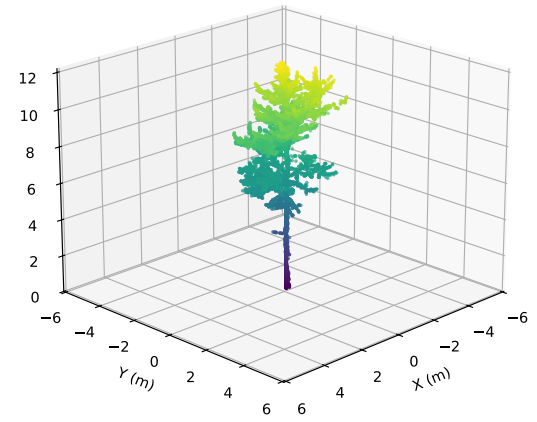
 doi:10.20944/preprints202604.1825.v1



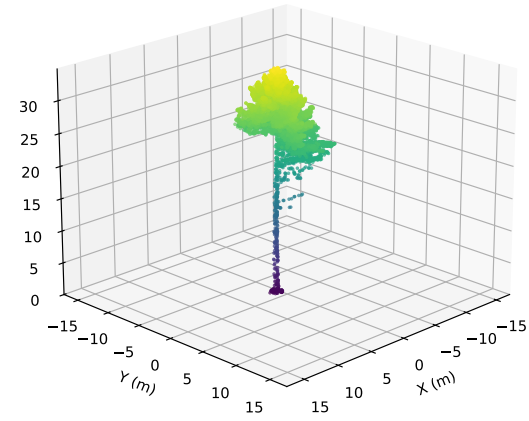
(**a**) Sycamore maple (*Acer pseudoplatanus*), a 19.43 m broadleaf captured via TLS



(**b**) Common hazel (*Corylus avellana*), a 9.84 m broadleaf captured via TLS



(**c**) Field Maple (*Acer campestre*), a 12.28 m broadleaf captured via TLS



(**d**) Douglas fir (*Pseudotsuga menziesii*), a 34.87 m conifer captured via ULS

Figure 1. Example tree point clouds from the FOR-species20K dataset illustrating variation in species, size, and acquisition platform. Each point cloud has been downsampled to 16,384 points as described in Section 2.4. Points are colored by height.

2.2. Flow Matching Framework

The goal of our generative model is to synthesize realistic 3D tree point clouds conditioned on a small set of observable attributes: species, acquisition platform, and tree height. Mathematically, we frame this as learning a transformation from an isotropic Gaussian distribution $p_0 = \mathcal{N}(0, I)$ to the normalized tree point cloud distribution $p_1 \approx p_{\text{data}}$, such that specifying only species, sensor type, and target height is sufficient to generate novel, structurally plausible individual tree point clouds. Figure 2 shows a conceptual diagram of this transformation. Our formulation treats individual points as independent samples during the generative process, with the permutation-equivariant neural network architecture (Section 2.3) responsible for learning and enforcing spatial relationships that yield coherent tree structures.

 doi:10.20944/preprints202604.1825.v1

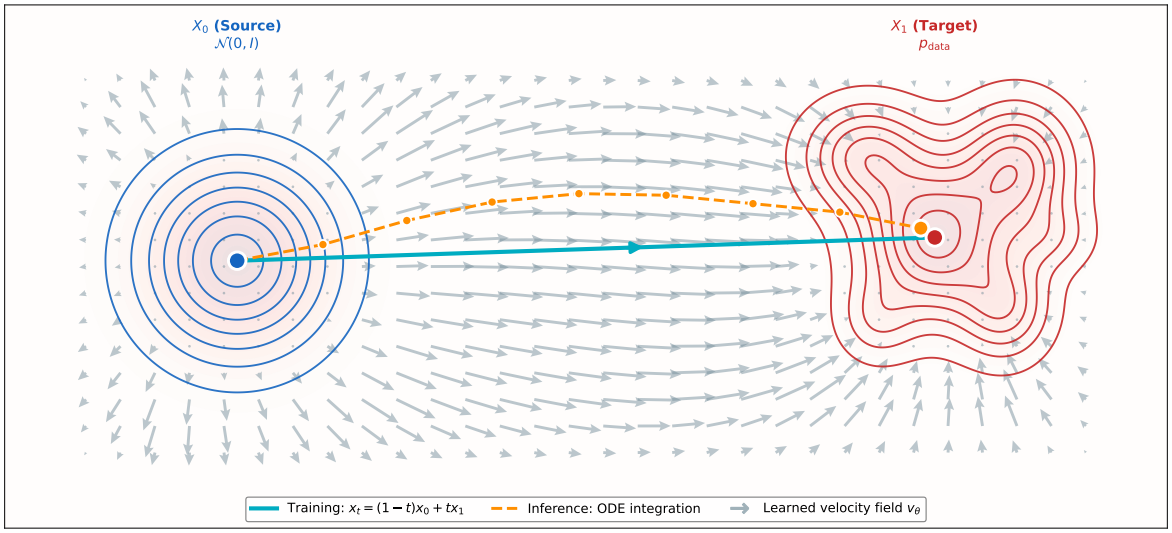


Figure 2. Conceptual illustration of the flow matching framework for 3D tree point cloud generation. **(Left)** Source distribution $p_0 = \mathcal{N}(0, I)$: points are sampled from a standard Gaussian distribution in $\mathbb{R}^3$. **(Center)** A neural network $v_\theta(x_t, t, c)$ learns a time-dependent velocity field that transports samples along the optimal transport path $x_t = (1-t)x_0 + tx_1$, where $t \in [0, 1]$. The heatmap illustrates the evolution of a 1D marginal density from a unimodal Gaussian at $t = 0$ to a bimodal distribution at $t = 1$. The conditioning vector c encodes tree attributes (species, acquisition platform, height). **(Right)** Target distribution $p_1 \approx p_{\text{data}}$: a normalized tree point cloud from the FOR-species20K dataset. At inference, synthetic trees are generated by sampling $x_0 \sim \mathcal{N}(0, I)$ and integrating the learned velocity field forward in time.

We adopt the Flow Matching framework, a simulation-free approach for training Continuous Normalizing Flows (CNFs) that has demonstrated strong performance in high-dimensional generative modeling [27,28]. The central concept is a time-dependent velocity field $v_t(x)$ that describes how points move through the data space over a continuous time interval $t \in [0, 1]$, effectively transporting probability mass from the source distribution (noise) to the target distribution (data).

We employ the Conditional Optimal Transport (OT) probability path, which defines a linear interpolation between noise and data as

$$x_t = (1-t)x_0 + tx_1 \tag{1}$$

where $x_0 \sim \mathcal{N}(0, I)$ is a sample from the source Gaussian and $x_1 \sim p_{\text{data}}$ is a sample from the target data distribution. The corresponding target velocity field that generates this path takes the form

$$u_t = x_1 - x_0 \tag{2}$$

The neural network v_θ is trained to predict this target velocity given the interpolated point x_t, the timestep t, and conditioning information c (encoding species, platform, and height). Training minimizes the Conditional Flow Matching (CFM) loss

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t, x_0, x_1}\left[\|v_\theta(x_t, t, c) - u_t\|^2\right] \tag{3}$$

where $t \sim \mathcal{U}[0, 1]$, $x_0 \sim p_0$, and $x_1 \sim p_{\text{data}}$. The CFM loss can be thought of as regressing the predicted velocity onto the target velocity using mean squared error. Importantly, this objective requires only samples from the source and target distributions, avoiding the expensive numerical simulation of differential equations during training.

 doi:10.20944/preprints202604.1825.v1

2.3. Model Architecture

We model v_θ as a transformer [40] with long skip connections following the U-ViT design [41], processing point clouds as sequences of coordinate tokens. Given an interpolated point cloud x_t, timestep t, and conditioning inputs c, the network outputs a per-point velocity vector $v_\theta(x_t, t, c) \in \mathbb{R}^3$ predicting the instantaneous direction of movement along the learned flow. Spatial information is encoded through fixed sinusoidal positional embeddings, while conditioning information in the form of timestep, species, platform, and height is provided as additional tokens prepended to the input sequence. Figure 3 shows a diagram of the model architecture, and full implementation details are provided below.

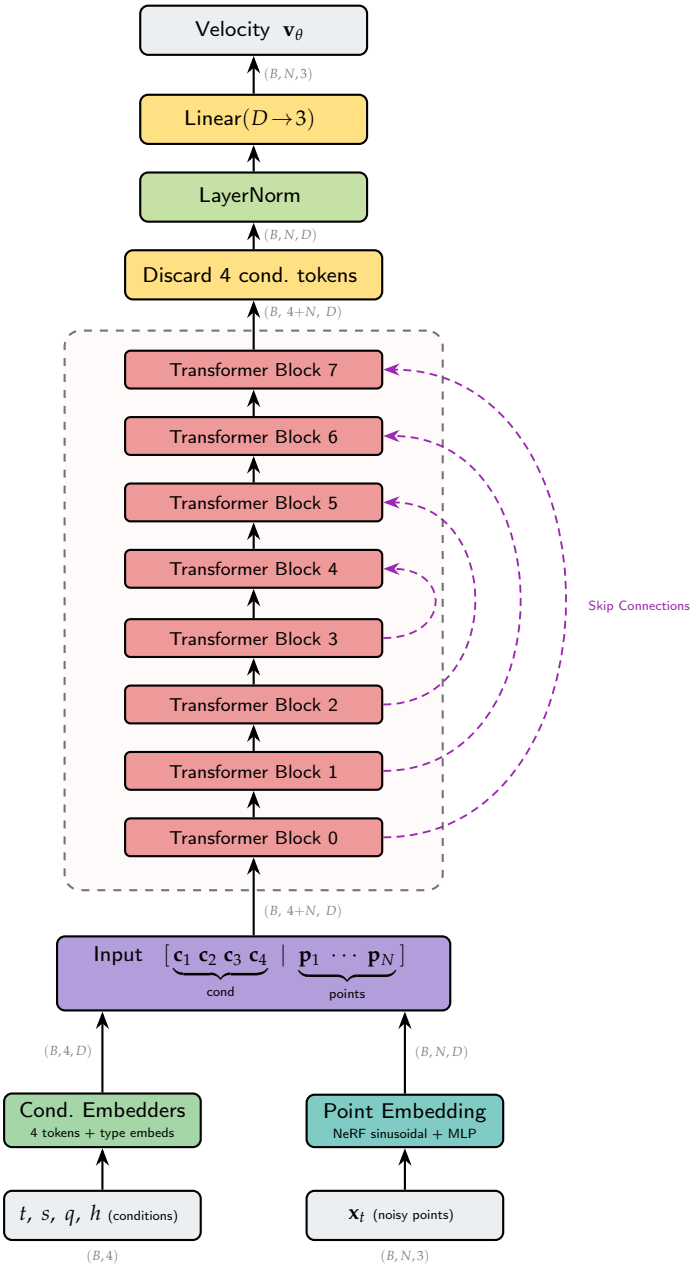


Figure 3. Architecture of the conditional flow matching network. Conditioning inputs are timestep t, species s, acquisition platform q, and tree height h; point coordinates p_a denote position along spatial axis a.

The input to the network consists of raw 3D point coordinates, which are transformed into high-dimensional token representations through sinusoidal positional encoding followed by a learned

 doi:10.20944/preprints202604.1825.v1

projection. We apply fixed NeRF-style frequency embeddings [42] independently to each spatial axis using the frequency series

$$f_k = 2^k \cdot \pi, \quad k = 0, 1, \ldots, L-1 \tag{4}$$

where L is the number of frequency bands. For each axis $a \in \{x, y, z\}$ and coordinate value p_a, the encoding concatenates sine and cosine components across all frequencies:

$$\text{enc}(p_a) = [\sin(p_a f_0),\ \cos(p_a f_0),\ \ldots,\ \sin(p_a f_{L-1}),\ \cos(p_a f_{L-1})] \tag{5}$$

The raw coordinates are concatenated with the axis-wise encodings to form a $(3 + 6L)$-dimensional feature vector for each point, which is then projected to model dimension d through a two-layer MLP with GELU activation. We use $L = 12$ frequency bands which gives us an initial data dimension of 75. Each of the N points in a point cloud is treated as an independent token in the input sequence.

The timestep $t \in [0, 1]$ is scaled to $[0, 1000]$ and encoded using sinusoidal embeddings with 256 frequency components following the standard transformer formulation [40], then projected through a two-layer MLP with SiLU activation to produce a d-dimensional vector. Species and acquisition platform are categorical variables encoded via learned embedding tables of dimension d. To enable classifier-free guidance (CFG) during inference (Section 2.5), each embedding table includes an additional null token at index $K + 1$, where K is the number of categories. Tree height, a continuous variable in $\mathbb{R}^1$, is first log-transformed, then projected to $\mathbb{R}^d$ through a separate two-layer MLP with SiLU activation. A learnable null embedding vector is also maintained for height to support CFG dropout of this continuous attribute. Each conditioning embedding is assigned a learned token-type embedding to distinguish its role, and the four resulting d-dimensional vectors are prepended to the point token sequence as separate tokens, forming an input sequence of length $4 + N$.

The transformer backbone consists of a sequence of pre-norm transformer blocks [40]. Each block contains two sub-layers: multi-head self-attention followed by a pointwise feedforward MLP with GELU activation and $4\times$ hidden dimension expansion. Both sub-layers use pre-LayerNorm residual connections. To prevent attention logit growth during training, we apply LayerNorm to the query and key projections independently per head before computing attention scores [43]. Conditioning information enters the transformer exclusively through self-attention between the prepended conditioning tokens and the point tokens, with no per-layer modulation of the network weights.

Following the U-ViT design [41], we add long skip connections between early and late transformer blocks. For $i < L/2$, the output of block i is concatenated with the output of block $L - 1 - i$ and projected back to model dimension d through a linear layer. These skip connections preserve the conditioning signal from early layers, where the prepended tokens have the most direct influence, through to the final layers of the network. Our implementation uses a model dimension of $d = 512$, 8 transformer blocks, and 8 attention heads.

After the transformer blocks, we append an output head that discards the four prepended conditioning tokens and operates only on the point tokens. A final LayerNorm is applied to the point token representations, followed by a linear projection mapping each d-dimensional token to $\mathbb{R}^3$, producing the per-point velocity prediction $v_\theta(x_t, t, c)$. The full model contains approximately 28.3 million trainable parameters.

2.4. Training

To enhance model robustness and promote geometric invariance, we employed data augmentation during training. Random rotation around the vertical (Z) axis was applied to each sample, and point order within each cloud was shuffled to enforce permutation invariance. Point clouds exceeding 16,384 points were randomly subsampled to this maximum count to manage computational load. We further employed a batch-level point sampling strategy in which a random target point count was drawn per batch from a power-law distribution $n = u^\alpha \cdot n_{\max}$, where $u \sim \mathcal{U}[0, 1]$ and $\alpha = 0.3$, with a floor of 256

 doi:10.20944/preprints202604.1825.v1

points. All point clouds within a batch were then subsampled to this target count, exposing the model to variable point densities while enabling efficient batched computation.

We optimized the model using AdamW with weight decay of 1×10^{-5}. Gradient clipping with a maximum norm of 1.0 was applied to prevent training instabilities arising from occasional large gradients. We employed mixed-precision training using bfloat16 via PyTorch's automatic mixed precision.

Because self-attention has $O(N^2)$ computational cost, training directly at the full point count of 16,384 is expensive. We therefore adopted a curriculum learning strategy that progressively increased the point count across three training stages, initializing each stage from the weights of the previous one. In the first stage, the model was trained at 4,096 points per cloud for 2,000 epochs with a learning rate of 1×10^{-4} and cosine annealing to 1×10^{-5}, using a per-GPU batch size of 32. In the second stage, the point count was increased to 8,192 for 1,000 epochs with a constant learning rate of 5×10^{-5} and a per-GPU batch size of 16. In the third stage, the point count was increased to 16,384 for 5,000 epochs with a learning rate of 5×10^{-5} and cosine annealing to 1×10^{-6}, using a per-GPU batch size of 8. At each transition, only the model weights were transferred; the optimizer and learning rate scheduler were reinitialized. All three stages drew training data from the same preprocessed dataset, which contains only trees with at least 16,384 points.

The training procedure at each stage was identical. A batch of normalized point clouds was sampled from the training set along with their associated conditioning labels (species, acquisition platform, and height). For each example in the batch, a random timestep t was drawn independently from $\mathcal{U}[0, 1]$, and Gaussian noise $x_0 \sim \mathcal{N}(0, I)$ was sampled with the same shape as the target point cloud x_1. Interpolated points x_t were then computed according to Equation 1. To enable classifier-free guidance during inference, conditioning inputs were randomly replaced with null embeddings with 15% probability. The model predicted velocities $v_\theta(x_t, t, c)$, and the Conditional Flow Matching loss in Equation 3 was computed against the target velocities u_t expressed by Equation 2.

Training was performed on four NVIDIA L40S GPUs using distributed data-parallel training via the Hugging Face Accelerate library. The implementation used PyTorch 3.10.19 with PyTorch version 2.8.0+cu129 and the `flow-matching` library [44] for optimal transport path computations. Source code is available at https://github.com/amarcozzi/TreeFlow.

2.5. Evaluation

Sample generation follows the standard flow matching inference procedure: integrating the learned velocity field forward in time to transport samples from the source distribution to the target distribution. Generation begins by sampling initial points $x_0 \sim \mathcal{N}(0, I)$ with point count matching that of the source tree. The learned velocity field $v_\theta(x_t, t, c)$ is then integrated from $t = 0$ to $t = 1$ using the Dormand-Prince adaptive ODE solver [45].

To improve sample fidelity to the conditioning vector, we apply classifier-free guidance (CFG) during inference [46]. At each integration step, the velocity is computed as

$$v = v_{\text{uncond}} + \omega \cdot (v_{\text{cond}} - v_{\text{uncond}}) \tag{6}$$

where v_{uncond} is the model output with conditioning information replaced by null embeddings, v_{cond} is the fully conditioned output, and ω is the guidance scale.

For each tree in the held-out test set, we generated 16 synthetic point clouds with CFG scale sampled independently for each sample from $\omega \sim \mathcal{U}[1.0, 4.5]$, exposing the evaluation to a range of conditioning strengths. Generated point clouds were then transformed from normalized coordinates back to metric scale by reversing the preprocessing described in Section 2.1. The source tree's point cloud was denormalized identically for all subsequent comparisons. We assess generation quality along two complementary axes, conditioning fidelity and distributional fidelity, reported globally, per genus, and per height bin at 5 m intervals from 0 to 40 m with a final bin for trees exceeding 40 m.

 doi:10.20944/preprints202604.1825.v1

Conditioning fidelity evaluates whether individual generated trees reproduce the geometric properties of their source trees, measured via six per-pair metrics. Chamfer distance measures the symmetric mean nearest-neighbor distance in meters between two point clouds. Because the model is trained with random rotation augmentation around the vertical axis, generated trees have no preferred horizontal orientation. We therefore align both clouds to a canonical orientation before comparison by centering at the mean, applying PCA to the xy-coordinates, and orienting the first principal axis so that the skewness of the projected coordinates is positive.

The next three metrics are derived from a stem-tracking algorithm that fits a polynomial spine to the trunk axis and computes cylindrical coordinates for each point relative to this spine, with radial distance measured perpendicular to the stem and arc length measured along it. The algorithm is described in Appendix B. The stem is divided into 30 equal arc-length slices, and the mean radial distance is computed within each slice. Maximum crown radius is the largest of these mean radial distances, measured in meters. Height at maximum crown radius is the arc-length position of the widest slice, converted to meters. Height to crown base is estimated by detecting the knee point in the cumulative radial profile, corresponding to the arc-length position where crown point density begins accumulating rapidly. The detection procedure is detailed in Appendix C.

The final two metrics compare the full spatial distribution of points within each tree. Vertical KDE divergence is the Jensen–Shannon divergence between one-dimensional kernel density estimates of the vertical coordinates evaluated at 64 equally spaced positions. This metric does not depend on the stem-tracking algorithm. Two-dimensional histogram divergence is the Jensen–Shannon divergence between 16×32 histograms of radial distance and arc length, with bin edges scaled to each tree's own range so that the comparison reflects distributional shape rather than absolute extent.

For each source tree, the six metrics are computed against all 16 of its generated samples and the median is retained as the per-tree summary statistic. Tables report the mean of these per-tree medians within each stratum, ensuring that every source tree contributes equally.

To aid interpretation, we construct two reference baselines from the real test set. The intra-class baseline pairs each test tree with 32 randomly sampled real neighbors that share its genus and height bin, representing the natural geometric variation among trees of comparable type and size. The inter-class baseline pairs each test tree with 32 neighbors drawn from different genera, representing the expected dissimilarity between unrelated trees. All six metrics are computed for these real–real pairs using the same alignment and feature extraction pipeline applied to generated samples.

Distributional fidelity evaluates whether the population of generated trees reproduces the statistical structure of the real population. We compute the Wasserstein-1 distance between the real and generated marginal distributions of three morphological properties: height at maximum crown radius, maximum crown radius, and height to crown base. For each property and stratum, the distance is computed between all real test trees in the stratum and all generated trees conditioned on sources within that stratum, requiring a minimum of five samples per group.

3. Results

Figure 4 illustrates the generative process for a single Douglas fir (*Pseudotsuga menziesii*, $h = 32.4$ m, ULS) across successive flow times from the isotropic Gaussian source at $t = 0$ to the final generated tree at $t = 1$. The integration reveals a coarse-to-fine hierarchy. At $t = 0.5$ the point cloud remains diffuse, exhibiting only a weak vertical elongation. By $t = 0.8$ a recognizable tree silhouette has formed, and between $t = 0.9$ and $t = 0.95$ the crown differentiates into a narrow conical profile characteristic of the species. Individual branch-scale features resolve only in the final stages of the flow. This progression suggests that the learned velocity field encodes tree structure at multiple spatial scales, allocating most of the transport budget to establishing gross geometry before committing to fine detail.

 doi:10.20944/preprints202604.1825.v1

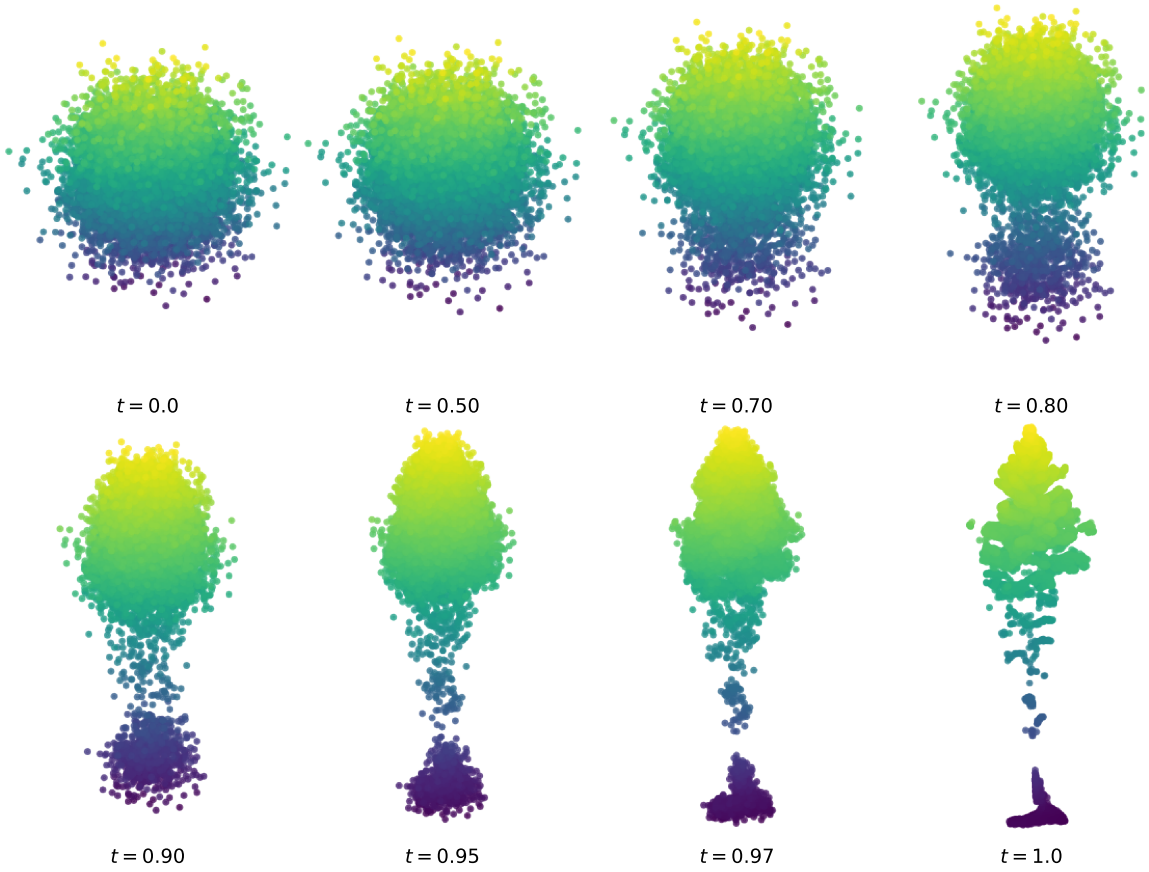


Figure 4. Time evolution of the generative process for a single Douglas fir (*Pseudotsuga menziesii*, $h = 32.4$ m, ULS) with classifier-free guidance scale $\omega = 3.0$. Each panel shows the point cloud at flow time t, progressing from the isotropic Gaussian source distribution ($t = 0$) to the final generated tree ($t = 1$) via integration of the learned velocity field v_θ. Time steps are spaced non-uniformly to emphasize the later stages of integration where fine structure emerges. Points are colored by height, and each panel is independently scaled to its spatial extent.

Figure 5 presents six real trees from the test set alongside four generated samples for each, spanning a range of species and tree sizes. Source trees were selected from the best-performing third of the test set ranked by median Chamfer distance, with height stratification to represent diverse size classes. The four displayed samples per source are those with the lowest Chamfer distance among the 16 generated. These represent favorable cases, and analogous figures for randomly selected trees are provided in the supplementary material. Within each row, all samples share the same conditioning inputs (species, height, and acquisition platform), so differences among the four samples reflect the model's learned variability rather than differences in input. The generated trees reproduce the overall crown shape, stem proportions, and vertical point density structure of their real counterparts while exhibiting variation in branch placement, crown extent, and lean. Given a fixed set of attributes, the model produces structurally distinct but morphologically plausible realizations of 3D tree architecture.

 doi:10.20944/preprints202604.1825.v1

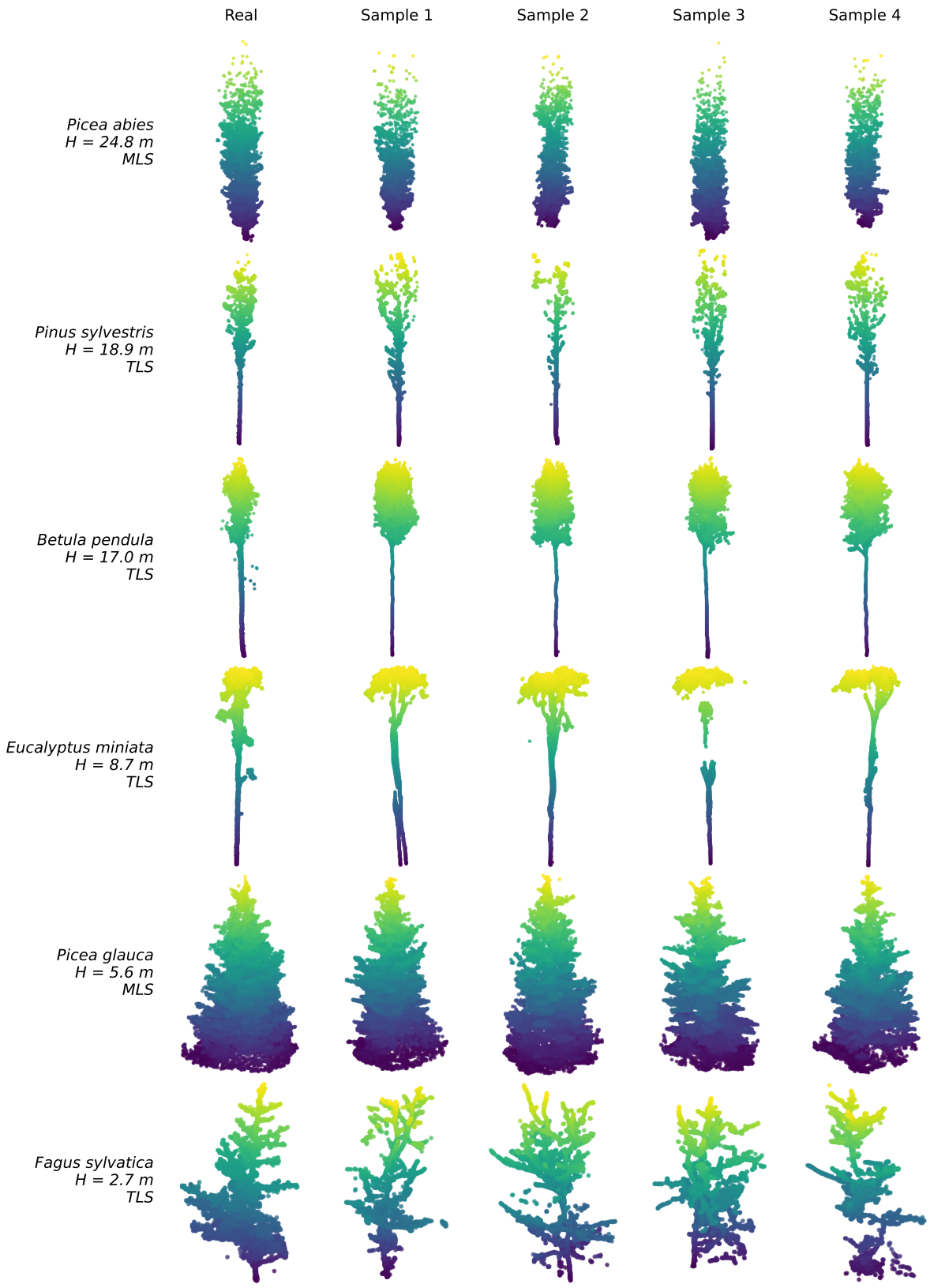


Figure 5. Six test-set trees (leftmost column) alongside four generated samples each, conditioned on the same species, height, and acquisition platform as the source. Source trees were drawn from the best-performing third of the test set as ranked by median Chamfer distance, stratified across height bins. Species, height, and platform are listed at left. Points are colored by height.

The model's response to the height conditioning variable is shown in Figure 6, which presents four generated Scots pine (*Pinus sylvestris*, TLS) point clouds at target heights of 5, 12, 20, and 30 m. Each

 doi:10.20944/preprints202604.1825.v1

sample was generated from independent Gaussian noise with only the target height varied. Species and platform were held fixed. The 5 m tree exhibits a broad, low-set crown typical of open-grown juveniles, while taller trees develop progressively longer bare stems and narrower, elevated crowns consistent with mature stand-grown individuals. These height-dependent structural relationships were not explicitly encoded and emerge entirely from the training data distribution.

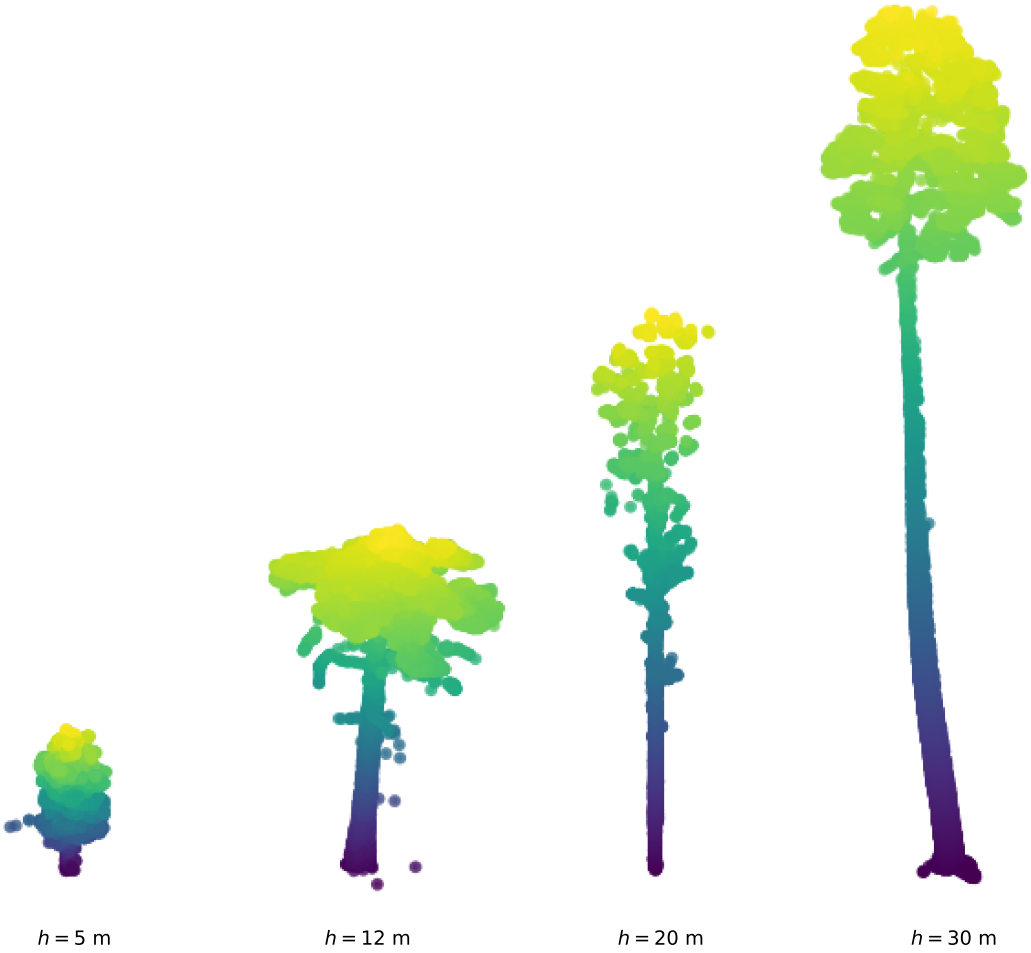


Figure 6. Generated Scots pine (*Pinus sylvestris*, TLS) point clouds at four target heights with classifier-free guidance scale $\omega = 3.0$. Each panel shows a single sample generated from independent Gaussian noise with only the target height h varied; species and platform are held fixed. Points are colored by height. All four panels share a common vertical scale and their trunk bases are aligned at $z = 0$, so the apparent size of each tree reflects its true generated height in meters and cross-panel comparison is direct.

Table 1 reports global conditioning fidelity across all 1,271 test trees using the six per-pair metrics described in Section 2.5, alongside the intra-class and inter-class reference baselines.

Table 1. Global conditioning fidelity across all 1,271 test trees, reported as the mean of per-tree medians. The intra-class baseline pairs each test tree with real neighbors sharing its genus and height bin; the inter-class baseline pairs each test tree with neighbors from different genera.

Metric	Generated	Intra-class	Inter-class
Chamfer distance (m)	0.581	0.559	1.025
|Δ| Height at max crown R (m)	2.882	2.932	8.377
|Δ| Max crown radius (m)	1.012	0.641	1.074
|Δ| Height to crown base (m)	2.020	2.113	5.568
Vertical KDE JSD	0.086	0.094	0.122
2D histogram JSD	0.240	0.251	0.319

 doi:10.20944/preprints202604.1825.v1

On five of the six metrics, generated samples perform at or near the intra-class baseline. The Chamfer distance of 0.581 m for generated trees compares to 0.559 m for the intra-class baseline, a difference of only 4%. This indicates that generated trees are approximately as geometrically similar to their source as two real trees of the same genus and height bin are to each other. Height at maximum crown radius difference (2.882 m vs. 2.932 m), height to crown base difference (2.020 m vs. 2.113 m), vertical KDE divergence (0.086 vs. 0.094), and 2D histogram divergence (0.240 vs. 0.251) all show generated samples matching or slightly outperforming the intra-class baseline. All six metrics fall substantially closer to the intra-class baseline than to the inter-class baseline, confirming that the model generates trees belonging to the correct morphological class.

The one metric where generated samples are notably worse than the intra-class baseline is maximum crown radius difference (1.012 m vs. 0.641 m), a 58% relative increase. This suggests that the model sometimes over- or under-estimates lateral crown extent even when the vertical structure of the tree is well captured.

The conditioning fidelity metrics in Table 1 assess individual-level accuracy but do not reveal whether the model introduces systematic biases across the population. We therefore compute the Wasserstein-1 (W_1) distance between the real and generated marginal distributions of three morphological properties. The W_1 distances for height to crown base (0.363 m) and height at maximum crown radius (0.777 m) are small relative to the range of tree heights in the dataset. These values indicate that the model places the crown base transition and widest crown section at statistically appropriate positions across the population. Maximum crown radius exhibits a substantially larger W_1 of 5.385 m, indicating a distributional shift in lateral crown extent that is not apparent from the per-pair conditioning fidelity metrics alone. To identify the source of this discrepancy, we stratify the evaluation by genus and height bin in the following analysis.

Table 2 stratifies the evaluation by genus, revealing substantial variation in generation quality across taxa. The strongest results appear for genera that are well represented in the training data and span moderate heights. *Pinus* ($n = 393$), the most abundant genus, achieves a Chamfer distance of 0.41 m and a crown radius W_1 of 0.24 m. *Quercus* ($n = 102$), *Tilia* ($n = 20$), and *Eucalyptus* ($n = 28$) show similarly strong conditioning fidelity, with Chamfer distances below 0.6 m and crown radius W_1 values under 1 m. *Euonymus* ($n = 5$), though represented by very few test trees, achieves the lowest Chamfer distance (0.20 m) and crown radius difference (0.17 m) of any genus. This likely reflects its compact, low-stature growth form.

Table 2. Conditioning fidelity and distributional fidelity stratified by genus. Genera are sorted alphabetically; n is the number of unique test trees. W_1 requires a minimum of five trees per stratum and is omitted where this threshold is not met.

		Conditioning fidelity						W_1 (m)		
Genus	n	CD	ΔHmCR	ΔCrR	ΔHCB	V-KDE	H2D	HmCR	CrR	HCB
Abies	9	0.81	4.49	0.76	3.84	0.080	0.233	2.09	0.73	2.00
Acer	124	0.69	2.30	0.69	1.43	0.050	0.252	0.85	0.32	0.57
Betula	36	0.54	2.76	0.64	1.68	0.103	0.274	1.06	0.38	0.88
Carpinus	66	0.58	2.03	0.93	1.43	0.067	0.290	0.79	1.04	0.29
Corylus	4	0.88	1.18	1.06	0.84	0.088	0.325	—	—	—
Crataegus	14	0.40	1.03	1.13	0.63	0.091	0.352	0.61	0.62	0.44
Eucalyptus	28	0.41	0.87	0.60	0.73	0.078	0.254	0.53	0.51	0.29
Euonymus	5	0.20	0.49	0.17	0.48	0.090	0.316	0.30	0.19	0.20
Fagus	199	0.76	4.04	1.22	2.88	0.072	0.257	0.53	2.20	0.40
Fraxinus	17	0.94	2.44	1.12	2.39	0.043	0.236	1.89	0.54	1.26
Larix	9	1.32	4.86	1.76	4.23	0.258	0.363	2.98	4.07	1.45
Picea	147	0.60	4.65	2.39	3.46	0.127	0.265	2.20	29.31	1.24
Pinus	393	0.41	2.22	0.53	1.50	0.079	0.195	0.88	0.24	0.27
Populus	14	0.66	2.11	1.21	1.29	0.104	0.322	0.78	0.98	0.74
Prunus	1	0.50	2.57	0.23	2.57	0.065	0.183	—	—	—
Pseudotsuga	64	0.98	5.24	1.54	3.59	0.222	0.298	2.16	26.69	1.50
Quercus	102	0.52	1.73	0.95	1.30	0.045	0.186	0.42	0.76	0.35
Tilia	20	0.39	1.47	0.56	0.76	0.059	0.261	0.45	0.16	0.32
Ulmus	19	0.39	1.94	0.48	0.87	0.063	0.312	1.53	0.41	0.36

Performance degrades for tall conifers and underrepresented genera. *Larix* ($n = 9$) has the highest Chamfer distance (1.32 m) and vertical KDE divergence (0.258), consistent with its sparse, open crown

 doi:10.20944/preprints202604.1825.v1

architecture being difficult to reconstruct from few training examples. *Picea* ($n = 147$) and *Pseudotsuga* ($n = 64$) exhibit crown radius W_1 values of 29.31 m and 26.69 m, respectively, far exceeding any plausible crown dimension. These extreme values indicate that either the model occasionally generates samples with unrealistic lateral extent, or the stem-tracking algorithm used to extract crown radius produces unreliable estimates for certain tree morphologies, or both. The stem tracker fits a polynomial spine to the trunk axis using density-weighted horizontal slices (Appendix B). Trees with strong lean, highly asymmetric crowns, or low branching can deflect the spine away from the true stem, inflating the radial coordinate for all points in that tree. Both *Picea* and *Pseudotsuga* are among the tallest genera in the dataset, suggesting that tree height may also contribute to the observed discrepancy.

Table 3 stratifies the evaluation by height bin, revealing a clear relationship between tree height and generation quality. Nearly every conditioning fidelity metric degrades monotonically with increasing height. Trees below 5 m achieve a Chamfer distance of 0.28 m and crown radius W_1 of 0.39 m. Trees in the 10–20 m range remain practical with Chamfer distances of 0.44–0.56 m. Above 25 m, the Chamfer distance approaches or exceeds 1.0 m and per-pair crown morphology metrics roughly double relative to the 15–20 m bin.

Table 3. Conditioning fidelity (mean of per-tree medians) and distributional fidelity (W_1, meters) stratified by height bin. Bins are 5 m intervals from 0 to 40 m with a final bin for trees exceeding 40 m; n is the number of unique test trees per bin.

		Conditioning fidelity						W_1 (m)		
Height bin	n	CD	ΔHmCR	ΔCrR	ΔHCB	V-KDE	H2D	HmCR	CrR	HCB
0–5 m	103	0.28	0.82	0.49	0.48	0.064	0.290	0.22	0.39	0.16
5–10 m	268	0.41	1.50	0.60	0.92	0.064	0.259	0.30	0.44	0.14
10–15 m	133	0.56	2.39	0.75	1.69	0.085	0.265	0.80	0.46	0.41
15–20 m	257	0.44	2.41	0.60	1.64	0.065	0.192	0.81	0.32	0.21
20–25 m	239	0.61	3.10	0.84	2.36	0.083	0.213	0.92	2.29	0.39
25–30 m	137	0.95	5.25	2.91	3.89	0.149	0.275	2.28	32.54	1.57
30–35 m	85	1.04	5.98	1.86	4.07	0.130	0.259	1.63	16.14	0.59
35–40 m	38	1.02	5.37	1.27	3.58	0.123	0.231	2.03	4.22	1.15
40+ m	11	0.92	5.97	1.07	3.91	0.092	0.187	3.89	0.55	2.21

The crown radius W_1 is moderate across all bins below 25 m (≤ 2.29 m) but spikes to 32.54 m in the 25–30 m bin and 16.14 m in the 30–35 m bin before returning to lower values above 35 m. This localization indicates that the extreme crown radius W_1 values identified at the genus level are concentrated in a narrow region of the height distribution rather than affecting tall trees uniformly. The 25–35 m range contains a mixture of tall conifers (*Picea*, *Pseudotsuga*) and tall broadleaves (*Fagus*) with diverse crown architectures. It is also the range where the stem-tracking algorithm is most susceptible to error due to long trunk segments and large crown extents. Disentangling model error from evaluation artifacts in this height range is not possible from the present analysis and is addressed in the Discussion.

An additional factor contributing to the height-dependent degradation is the normalization procedure described in Section 2.1. Because all point clouds are scaled by tree height, a tall tree occupies the same normalized volume as a short tree but distributes its points across a proportionally larger metric-scale structure. At fixed point count, this results in lower effective spatial resolution for taller trees. The model must compensate by learning coarser representations of fine-scale crown features for these larger trees.

4. Discussion

The results presented here demonstrate that a conditional flow matching model with a general transformer architecture can generate structurally plausible 3D tree point clouds from species, acquisition platform, and tree height alone, bypassing the chained submodel pipelines that have traditionally been required to estimate 3D structure from inventory attributes. On five of six conditioning fidelity metrics, generated trees perform at or near the intra-class baseline, and the model produces morphologically distinct samples that respect species-specific crown architecture and scale appropriately with

 doi:10.20944/preprints202604.1825.v1

height. We obtained these results using a single dataset, a single architecture with no domain-specific structural priors, and a minimal set of conditioning variables. We chose this minimal configuration to establish a baseline that future work can extend through expanded training data, richer conditioning information, and alternative model architectures.

The dominant pattern across both the genus-level and height-level stratification is that generation quality degrades with increasing tree height. This degradation has a clear mechanistic explanation. Because all point clouds are scaled by tree height during normalization, a 35 m tree distributes the same number of points across a proportionally larger metric-scale structure than a 5 m tree, reducing the effective spatial resolution available to the model. While we acknowledge the deficiency of this approach, this normalization scheme allowed for efficient training of a general architecture as described above, and is something that can be explored in future studies. In practice, the model produces its strongest results for trees below approximately 25 m, where Chamfer distances remain below 0.7 m and distributional metrics are well-behaved across most genera.

Maximum crown radius is the one metric where generated samples consistently underperform the intra-class baseline, with per-pair differences 58% larger at the global level and distributional W_1 values of 29.31 m for *Picea* and 26.69 m for *Pseudotsuga*, far exceeding any plausible crown dimension. These extreme values are concentrated in the 25–35 m height bins rather than affecting tall trees uniformly. The first contributing factor is an evaluation artifact. The stem-tracking algorithm fits a polynomial spine to the trunk axis, and trees with strong lean, asymmetric crowns, or low branching can deflect the spine away from the true stem, inflating the radial coordinate for all points. Both *Picea* and *Pseudotsuga* are among the tallest genera in the dataset, and the stem-tracking algorithm is most susceptible to spine deflection when trunk segments are long and crown extents are large. The second factor is a more fundamental limitation of the current dataset. Lateral crown extent is influenced by competition, light environment, site quality, and developmental history, none of which are captured by species and height alone. The model may be producing reasonable samples from the distribution it has learned, but that distribution is broad because the conditioning variables do not sufficiently constrain it.

The model learns to approximate a distribution over 3D point clouds spanned by the training data and cannot generalize to species, crown morphologies, or geographic contexts outside that distribution. While the evaluation demonstrates strong performance on the held-out test set, the train and test splits were stratified by species and height across the same underlying datasets, and we can expect reasonable similarity between the two distributions. The FOR-species20K dataset is geographically concentrated in European forests with imbalanced platform representation (Section 2.1), and the per-genus results in Table 2 confirm that generation quality correlates with sample size in the training set. Expanding the training dataset is the most direct path to broader applicability. There are many single-tree and plot-level TLS datasets across the globe that could extend the geographic, species, and crown morphology coverage of the existing benchmark. Examples include the 3Dtrees.earth[1] and Global TLS[2] datasets. The Interagency LiDAR Monitoring and Research Applications (IntELiMon) is another possible dataset consisting of many single scan plots which can be used for either individual tree training data, or a modified approach in which we train a model to learn the conditional generation of single scan plots rather than individual trees [47].

Another limitation of this modeling approach is that the transformer architecture imposes $O(N^2)$ computational complexity in the self-attention mechanism, which is the primary bottleneck for scaling to larger point counts. At 16,384 points with FlashAttention, training on L40S GPUs and inference via adaptive ODE solving are tractable, but scaling to the full point densities typical of TLS scans, often exceeding 100,000 points per tree, is not feasible with the current approach. The multiple forward passes required by classifier-free guidance at each ODE solver step further compound this cost. We investigated several sparse self-attention techniques to address the quadratic scaling but found that

[1] https://3dtrees.earth/
[2] https://www.global-tls.net/

 doi:10.20944/preprints202604.1825.v1

custom implementations were slower than full self-attention with FlashAttention for the point counts considered here.

A related concern is whether the general transformer architecture is the right choice for this problem. Full self-attention means that every point attends to every other point, including pairs with no meaningful spatial relationship, such as a stem point and a distant terminal branch point. An architecture with a stronger inductive bias for 3D structure, such as point transformer networks or octree representations, could potentially reduce computation by restricting attention to local neighborhoods while still capturing the long-range dependencies needed for global crown shape [48,49]. A second promising direction is latent-space generation, in which an encoder compresses the point cloud to a fixed-size representation and the flow matching model operates in that lower-dimensional space [50]. This would decouple the token count from the point count, and the idea has shown strong results in image generation [51] and point cloud generation [30]. An ablation study comparing generation quality and computational cost across these architectural families is an important direction for future work.

Species and height alone do not capture the range of factors that influence crown morphology, including competition, light environment, site quality, and developmental history. As noted above, this is likely a contributing factor to the crown radius underperformance observed for several genera. The most immediate path to richer conditioning is the inclusion of variables that can be computed directly from the training data using the methods presented in this paper, such as crown base height, maximum crown radius, and diameter at breast height. Beyond tree-level attributes, conditioning on partial 3D information such as a low resolution ALS scan or a 2D canopy height model could enable the generation of 3D structure at landscape scales [38]. The freely available CONUS-scale NAIP-CHM data product would make such an approach feasible for open canopies across the continental United States [21].

With the conditioning expansion described above, TreeFlow could replace the crown construction module in FastFuels, which currently builds volumetric tree crowns from species, height, diameter, and crown base height using parametric crown profile models and probabilistic mass distribution functions [1]. These submodels are weakly parameterized for many species and rely on idealized geometric envelopes. TreeFlow offers two paths: generated point clouds can replace the profile and mass distribution chain directly, or generated samples can be used to fit the parameters of the existing functions. The framework also extends naturally as richer training data become available. Per-point semantic labels from datasets such as SegmentedForests [52] would allow joint generation of 3D positions and component classes (stem, branch, foliage), and methods that estimate biomass directly from TLS scans [53] suggest the possibility of generating voxels with mass quantities rather than assigning them through downstream allometry.

More broadly, the generated point clouds can serve as the individual-tree structural component for forest digital twins, which have been proposed as integrative modeling frameworks but currently lack scalable methods for populating 3D tree geometry from inventory data [8,54]. The same structure can support radiative transfer simulation [3,4], canopy light interception modeling [5], and quantitative structure modeling [55].

5. Conclusions

In this paper we presented TreeFlow, a conditional flow matching model that generates 3D individual tree point clouds from species, acquisition platform, and tree height, trained entirely on real laser scanning data from the FOR-species20K benchmark dataset. On five of six conditioning fidelity metrics, generated trees perform at or near the intra-class baseline, indicating that they are approximately as similar to their source trees as two real trees of the same genus and height class are to each other. The model captures species-specific crown architecture and produces morphologically distinct samples that scale appropriately with height, all without domain-specific structural priors or chained submodel pipelines. Generation quality is strongest for trees below 25 m and degrades at

 doi:10.20944/preprints202604.1825.v1

greater heights due to reduced effective spatial resolution under the current normalization scheme, while maximum crown radius remains the least well-constrained attribute across genera and height classes.

The minimal conditioning configuration used here establishes a baseline that can be extended in several directions: expanding the training data to include additional geographic regions and species, enriching the conditioning vector with variables such as crown base height and diameter at breast height, and exploring architectures with stronger inductive biases for 3D structure or latent-space generation to scale beyond the current 16,384-point limit. With these extensions, TreeFlow could serve as a data-driven alternative to parametric crown construction in platforms such as FastFuels, and more broadly as a scalable method for populating 3D tree geometry in forest digital twins.

Funding: This research was funded by Department of Defense, Environmental Security Technology Certification Program, grant numbers RC23-7626, RC24-8162, and NH25-9025

Data Availability Statement: All source code for data preprocessing, model training, sample generation, figure creation, and evaluation is provided in the GitHub repository: https://github.com/amarcozzi/TreeFlow. Model checkpoints, training validation, final weights, and all generated samples are available in the Zenodo repository: https://zenodo.org/records/19719862.

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments). Where GenAI has been used for purposes such as generating text, data, or graphics, or for study design, data collection, analysis, or interpretation of data, please add "During the preparation of this manuscript/study, the author(s) used [tool name, version information] for the purposes of [description of use]. The authors have reviewed and edited the output and take full responsibility for the content of this publication."

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Additional Results

Appendix A.1. Classifier-Free Guidance Scale Sensitivity

As described in the Evaluation Section, each of the 16 synthetic point clouds generated per test tree used a classifier-free guidance scale ω drawn independently from a uniform distribution. This design allows us to examine how guidance strength affects generation quality post hoc without requiring additional inference runs.

Figure A1 shows each of the six conditioning fidelity metrics as a function of ω. Within each panel, the solid line traces the binned median across all real–generated pairs falling in that ω bin, the shaded band spans the interquartile range, and the dashed horizontal line marks the corresponding intra-class reference baseline reported in Table 1.

All six metrics remain close to the intra-class baseline across the sampled guidance range, indicating that generation quality is not strongly sensitive to the choice of ω within this interval. Chamfer distance, the three morphological difference metrics, and the vertical KDE divergence show essentially flat medians with consistent spread. The 2D histogram divergence exhibits a mild upward trend with increasing ω, suggesting that stronger guidance may slightly reduce internal structural diversity relative to the source tree. The interquartile range is approximately constant across ω for every metric, indicating that guidance strength primarily affects central tendency rather than the variability of generation quality. These results suggest that the model produces robust outputs across a broad range of guidance strengths, and that practitioners can select ω within this range without substantial impact on fidelity.

 doi:10.20944/preprints202604.1825.v1

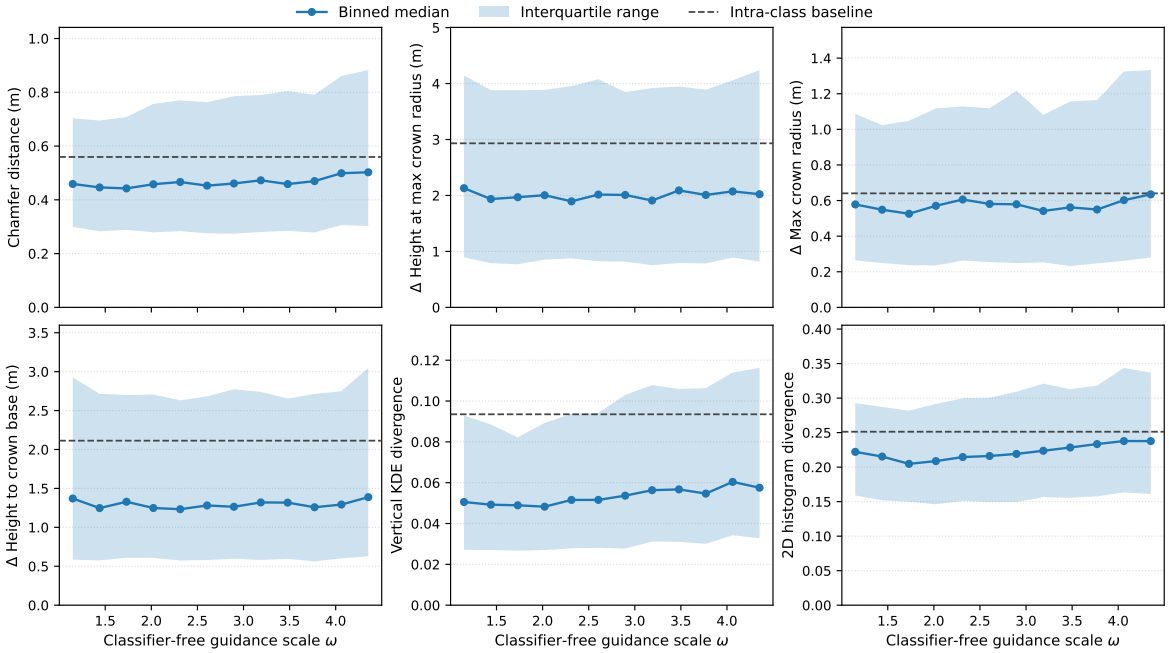


Figure A1. Conditioning fidelity metrics as a function of the classifier-free guidance scale ω. Each panel shows one of the six per-pair metrics from Table 1. The solid line is the median over all real–generated pairs within each ω bin, the shaded band spans the interquartile range, and the dashed horizontal line marks the intra-class baseline from Table 1. All six metrics remain close to the intra-class baseline across the sampled range, with only the 2D histogram divergence exhibiting a mild upward trend.

Appendix A.2. Samples

To complement the favorable cases shown in Figure 5, we provide three additional sets of qualitative samples drawn from different regions of the quality distribution. Figure A2 shows source trees from the middle third of the test set ranked by median Chamfer distance, representing typical generation quality. Figure A3 shows source trees from the bottom third and includes several obvious failure cases. Figure A4 shows a random selection across the full test set. As in Figure 5, all four samples within a row share the same species, height, and acquisition platform, so variation across samples reflects the model's learned variability rather than differences in conditioning.

 doi:10.20944/preprints202604.1825.v1

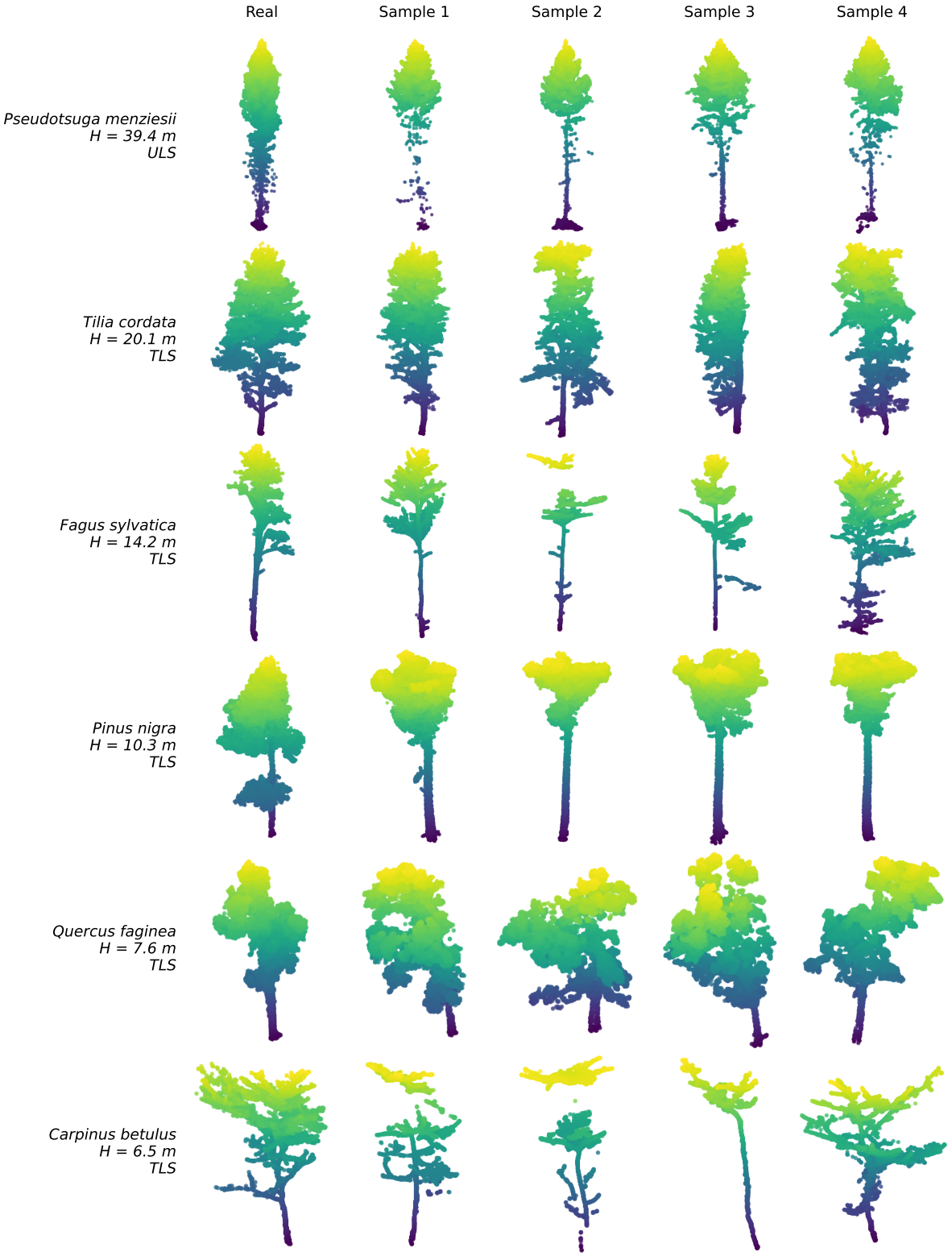


Figure A2. Representative generation quality. Six test-set trees (leftmost column) alongside four generated samples each, conditioned on the same species, height, and acquisition platform as the source. Source trees were drawn from the middle third of the test set as ranked by median Chamfer distance and reflect typical model performance. Species, height, and platform are listed at left. Points are colored by height.

 doi:10.20944/preprints202604.1825.v1

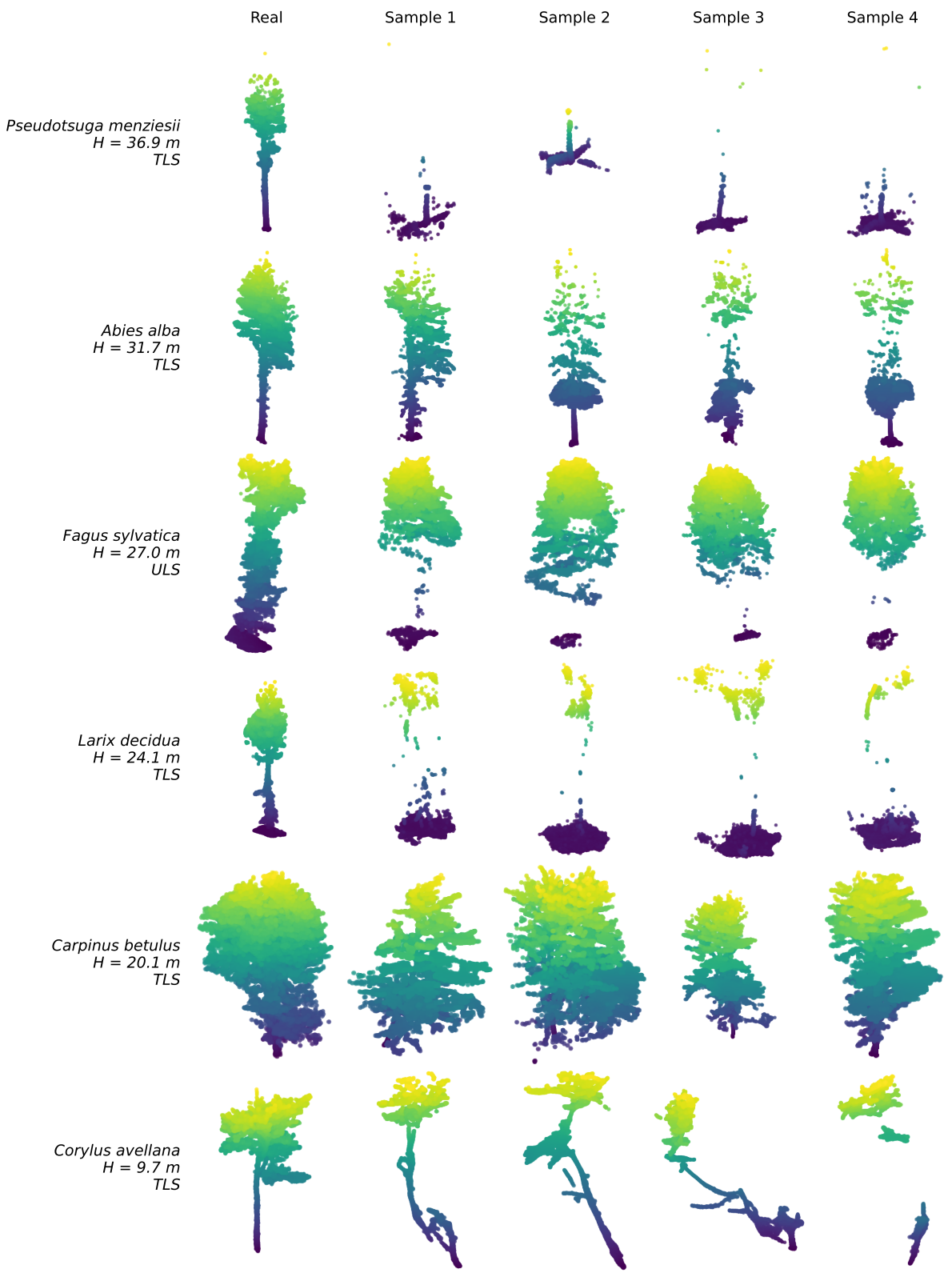


Figure A3. Worst case generation quality. Six test-set trees (leftmost column) alongside four generated samples each, conditioned on the same species, height, and acquisition platform as the source. Source trees were drawn from the bottom third of the test set as ranked by median Chamfer distance and illustrate the poorest generation quality observed in our analysis, including missing stems, fragmented crowns, and gross morphological errors. Species, height, and platform are listed at left. Points are colored by height.

 doi:10.20944/preprints202604.1825.v1

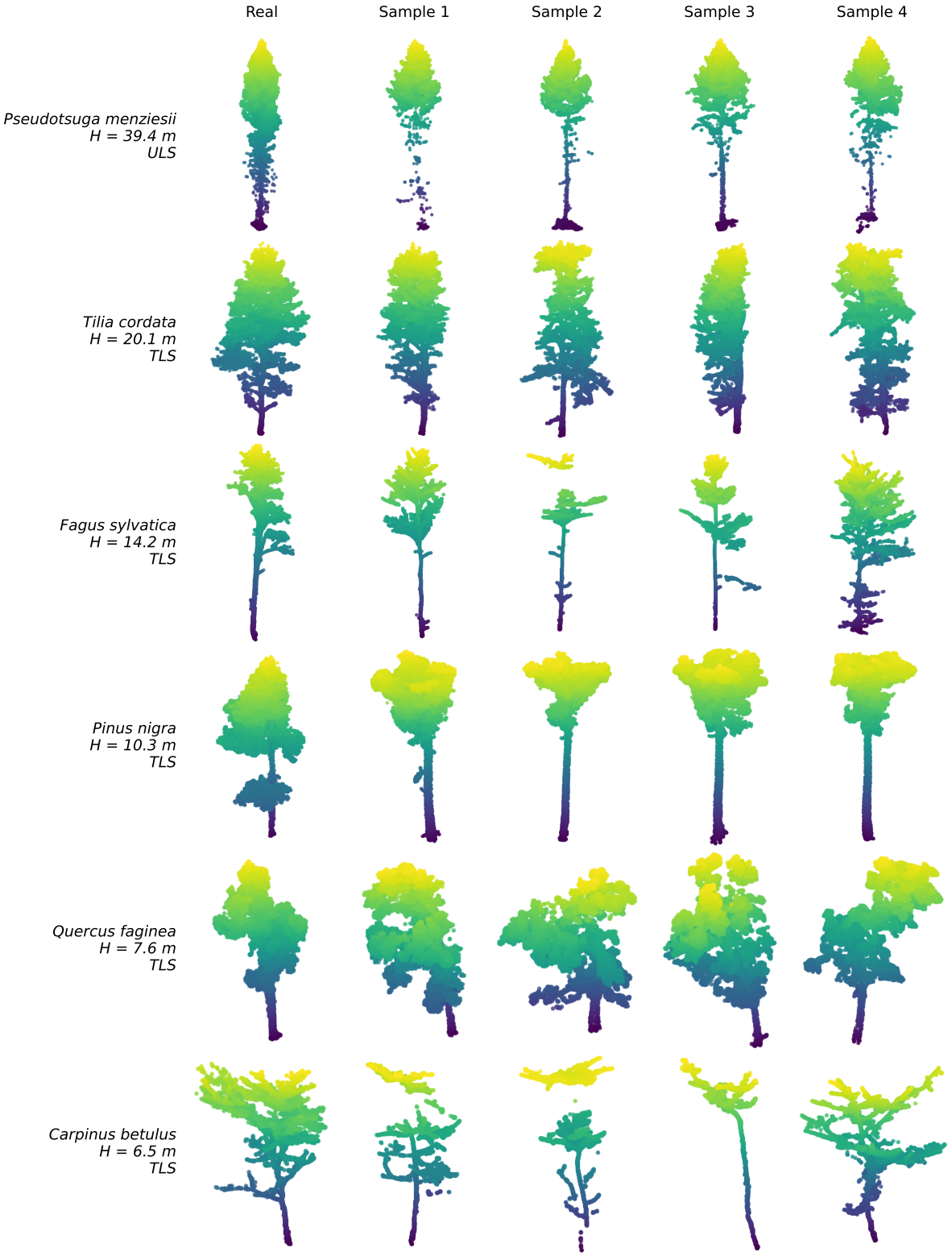


Figure A4. Random samples. Six test-set trees (leftmost column) alongside four generated samples each, conditioned on the same species, height, and acquisition platform as the source. Source trees were drawn at random from the test set, so the displayed samples reflect the full range of generation quality without selection bias. Species, height, and platform are listed at left. Points are colored by height.

Appendix B. Stem Tracking Algorithm

The stem-tracking algorithm estimates the central axis of the trunk and main stem from an unstructured 3D point cloud, then expresses each point in cylindrical coordinates relative to this axis. The algorithm proceeds in five stages: trunk base detection, bottom-up spine tracking, iterative

 doi:10.20944/preprints202604.1825.v1

refinement, outlier rejection, and polynomial fitting with arc-length parameterization. All parameter values reported below were held fixed across all trees in the dataset.

Stage 1: Trunk Base Detection

The first stage identifies the xy-position of the trunk base, which serves as the starting point for upward tracking. A naive approach would select the densest horizontal cluster in a thin slab near the ground, but low branches that extend below the base of the live crown can produce clusters that are denser than the trunk itself, pulling the estimate away from the true stem position.

We instead exploit the fact that the trunk persists vertically through many horizontal slices while a branch appears in only one or two. The lower 30% of the tree by height is isolated as the base region and divided into six horizontal sub-slices. Within each sub-slice, a mean-shift procedure with Gaussian kernel bandwidth equal to 15% of the base region's horizontal extent converges to the local density maximum, producing one candidate xy-position per sub-slice. Each candidate is then scored by its vertical persistence: the number of sub-slices that contain at least one point within the kernel bandwidth of that candidate. The candidate with the highest persistence score is selected as the trunk base position. Ties are broken in favor of the candidate closest to the horizontal median of the base region.

Stage 2: Bottom-Up Spine Tracking

Beginning from the trunk base position identified in Stage 1, the algorithm tracks the stem center upward through 20 horizontal slices spanning the full height of the tree. Within each slice, a weighted centroid of the horizontal coordinates is computed using two sources of information, while the vertical coordinate of the spine position is taken as the median height of the points in the slice.

The first source of information is spatial proximity to the previous slice center. A Gaussian kernel assigns higher weight to points near the expected stem position, with standard deviation equal to 10% of the current slice's horizontal extent. To prevent the kernel from growing excessively wide in canopy slices where the horizontal extent is large, the standard deviation is capped at twice the value computed for the base region.

The second source is local point density. For each point in the slice, the mean distance to its 16 nearest horizontal neighbors is computed. Points in tight clusters, as typically found along the trunk, receive higher weight than isolated points in the outer canopy.

The two sources are combined as follows. Let $(\hat{x}, \hat{y})$ denote the spine center from the previous slice, σ the capped Gaussian standard deviation, and $\bar{d}_i^{(k)}$ the mean distance from point i to its $k = 16$ nearest horizontal neighbors. The proximity and density weights for each point i in the slice are

$$w_i^{\text{prox}} = \exp\left(-\frac{(x_i - \hat{x})^2 + (y_i - \hat{y})^2}{2\sigma^2}\right), \qquad w_i^{\text{dens}} = \frac{1}{\bar{d}_i^{(k)}} \Big/ \max_j \frac{1}{\bar{d}_j^{(k)}} \tag{A1}$$

The density weights are normalized by their maximum value so that the two terms contribute on comparable scales. The combined weight is $w_i = w_i^{\text{prox}} \cdot w_i^{\text{dens}}$, and the horizontal spine position for the slice is the weighted mean of the xy-coordinates.

To prevent the tracker from jumping to a distant branch between consecutive slices, the horizontal displacement from the previous spine position is clamped to a maximum of 5% of the total tree height per step.

Stage 3: Iterative Refinement

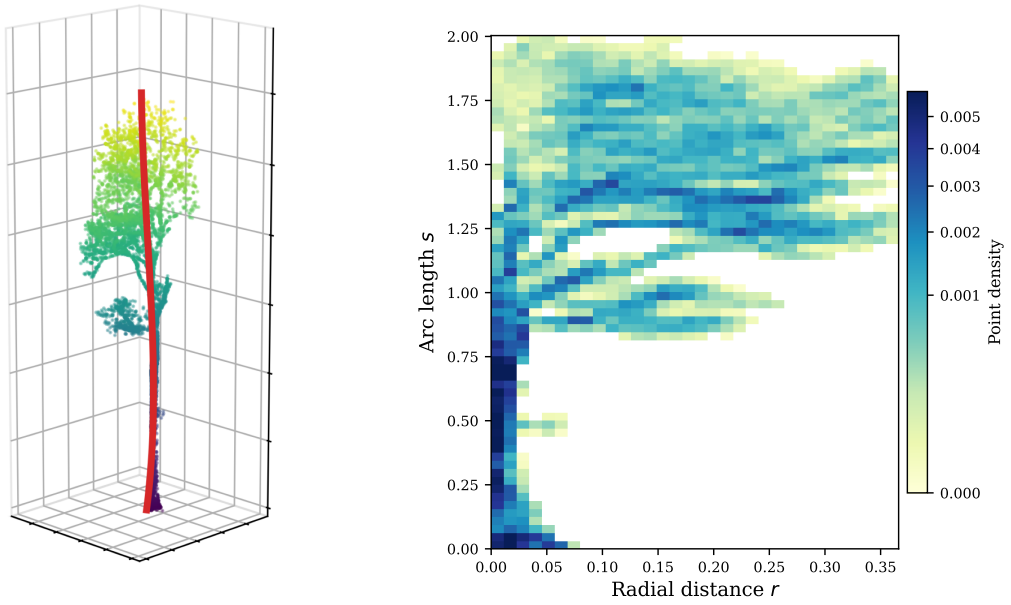
The spine positions from Stage 2 are smoothed by fitting cubic polynomials $p_x(z)$ and $p_y(z)$ to the tracked xy-positions as functions of height z. The polynomial curves are then evaluated at the midpoint of the lowest height bin to produce a refined starting position, and the bottom-up tracking

 doi:10.20944/preprints202604.1825.v1

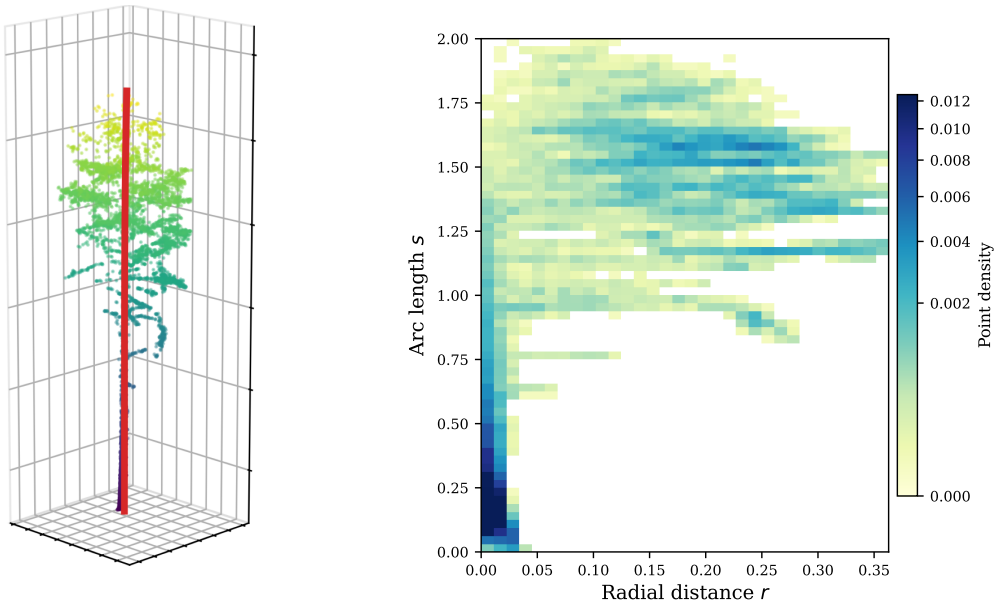
pass of Stage 2 is repeated once using this new initialization. This refinement corrects cases where the initial base estimate was slightly offset, which can bias the tracked spine through all subsequent slices.

Stage 4: Outlier Rejection

Before the final polynomial fit, spine positions that deviate strongly from the overall trend are removed. A degree-1 polynomial is fit to the tracked positions, and the Euclidean residuals in the horizontal plane are computed. Positions with residuals exceeding the median residual by more than 2.5 times the median absolute deviation are discarded. This step removes spine positions that were pulled toward isolated canopy clusters despite the clamping constraint in Stage 2.



(**a**) Stem-tracking output for a 30.1 m narrow-leaved ash (*Fraxinus angustifolia*) captured via TLS.



(**b**) Stem-tracking output for a 15.2 m Scots pine (*Pinus sylvestris*) captured via ULS.

Figure A5. Left: Three-dimensional point cloud colored by height with the fitted polynomial spine shown in red. **Right:** Two-dimensional density histogram of the cylindrical coordinates (r, s) computed relative to the spine using 16×32 bins.

 doi:10.20944/preprints202604.1825.v1

Stage 5: Polynomial Fitting and Arc-Length Parameterization

Cubic polynomials $p_x(z)$ and $p_y(z)$ are fit to the remaining spine positions, defining a smooth parametric curve $(p_x(z),\ p_y(z),\ z)$ that represents the central stem axis. For each point in the original cloud, the radial distance r is computed as the perpendicular distance to the spine at the corresponding height:

$$r_i = \sqrt{(x_i - p_x(z_i))^2 + (y_i - p_y(z_i))^2} \tag{A2}$$

Rather than using raw height z as the second cylindrical coordinate, we parameterize position along the spine by arc length s. For a curved or leaning stem, equal increments in z do not correspond to equal distances along the stem, and arc length provides a more physically meaningful measure of position. The arc-length differential is

$$\frac{ds}{dz} = \sqrt{p_x'(z)^2 + p_y'(z)^2 + 1} \tag{A3}$$

where p_x' and p_y' are the derivatives of the fitted polynomials. We evaluate this expression on a uniform grid of 500 points along z and compute the cumulative arc length via the trapezoidal rule. Each point's arc-length coordinate is then obtained by linear interpolation into this cumulative grid.

The resulting coordinates (r_i, s_i) for each point provide the basis for the maximum crown radius, height-to-crown-base, and two-dimensional histogram metrics described in Section 2.5.

Appendix C. Height to Crown Base Algorithm

Height to crown base is estimated from the cylindrical coordinates (r_i, s_i) produced by the stem-tracking algorithm described in Appendix B. The key observation is that for most tree architectures, moving upward along the stem from the base, the mean radial distance of points from the stem remains small through the bare trunk region and then increases sharply once the crown begins. The height to crown base corresponds to the arc-length position where this transition occurs.

The arc-length axis is divided into 30 equal slices from $s = 0$ to $s = s_{\max}$, and the mean radial distance $\bar{r}_k$ is computed for each slice k. The cumulative radial profile is then constructed as

$$C_k = \sum_{j=1}^{k} \bar{r}_j \tag{A4}$$

For a tree with a distinct bare stem below the crown, C_k increases slowly through the trunk region and then accelerates once the crown is reached, producing a concave curve when plotted against arc-length position.

To detect the transition point, we apply a modified version of the Kneedle algorithm [56], which identifies knee points in concave or convex curves by maximizing the distance between the curve and a reference line. Both the arc-length positions and the cumulative profile are first normalized to the unit interval:

$$\hat{s}_k = \frac{s_k - s_1}{s_K - s_1}, \qquad \hat{C}_k = \frac{C_k - C_1}{C_K - C_1} \tag{A5}$$

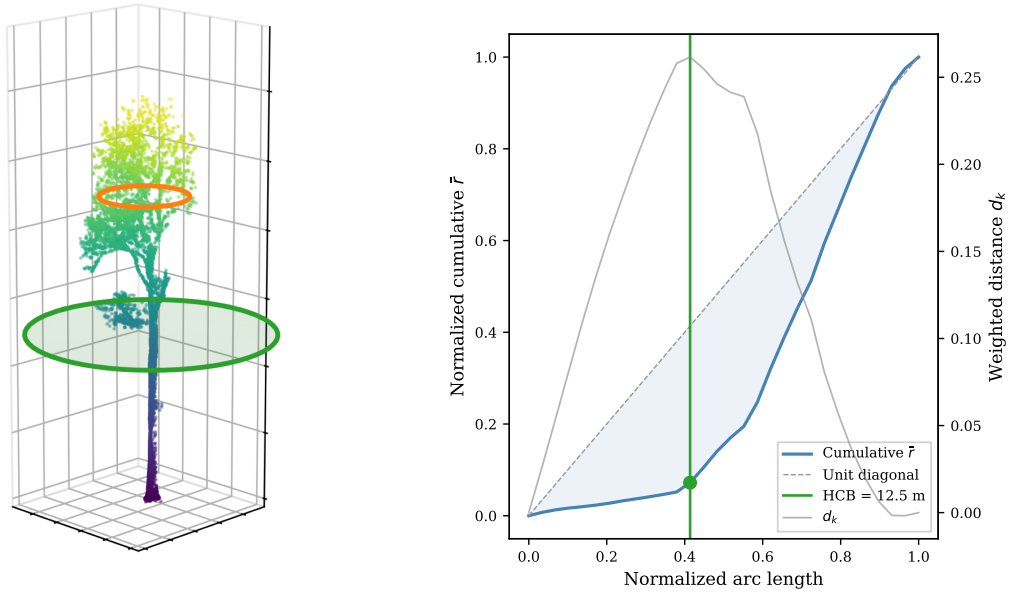
where K is the number of slices with at least one point. In the standard Kneedle formulation, the knee is the point of maximum perpendicular distance from the diagonal $\hat{C} = \hat{s}$. We modify the detection function by introducing a tapering weight that biases the detection toward lower positions on the stem:

$$d_k = (\hat{s}_k - \hat{C}_k) \cdot (1 - \hat{s}_k)^{1/2} \tag{A6}$$

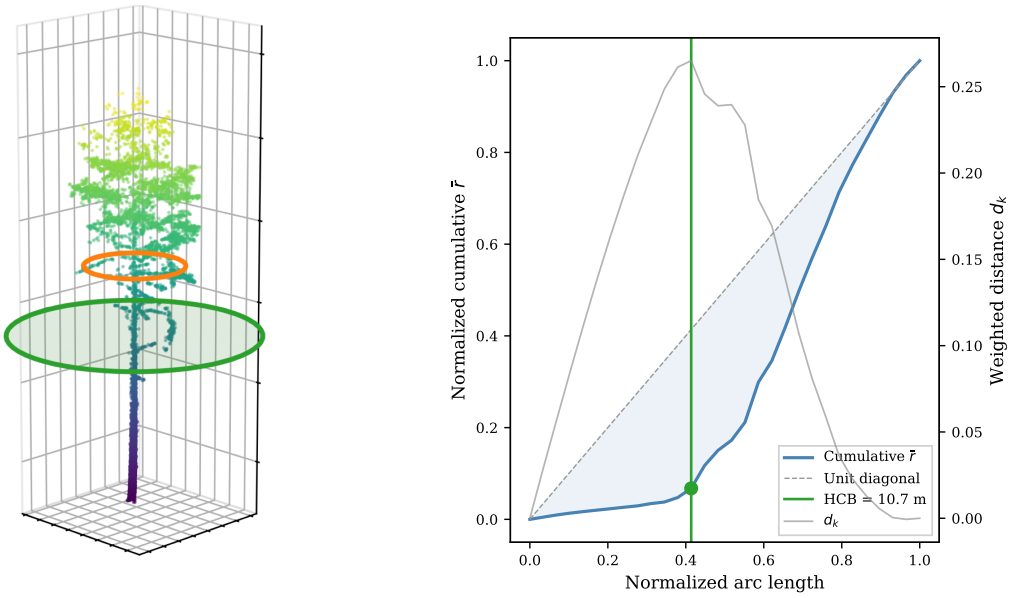
The first term $(\hat{s}_k - \hat{C}_k)$ is largest where the cumulative curve lags farthest behind the diagonal, which corresponds to the region where radial extent is just beginning to accumulate. The second term $(1 - \hat{s}_k)^{1/2}$ progressively downweights positions higher on the stem, preventing the algorithm from selecting spurious knee points in the upper crown where the cumulative curve may plateau or fluctuate.

 doi:10.20944/preprints202604.1825.v1

This modification reflects the prior expectation that the true crown base occurs in the lower portion of the tree. The crown base is assigned to the slice $k^* = \arg\max_k d_k$, and the height to crown base in meters is computed as $\mathrm{HCB} = (s_{k^*}/s_{\max}) \cdot h$, where h is the tree height.



(**a**) Narrow-leaved ash (*Fraxinus angustifolia*), H = 30.1 m, captured via TLS. The crown base is detected at 12.5 m along the stem.



(**b**) Scots pine (*Pinus sylvestris*), H = 15.2 m, captured via ULS. The crown base is detected at 10.7 m along the stem.

Figure A6. Height-to-crown-base detection on two trees with contrasting architecture. Left panels show the three-dimensional point cloud with the fitted polynomial spine (red), detected crown base plane (green ellipse), and maximum crown radius ring (orange ellipse). Right panels show the normalized cumulative radial profile (blue curve), the unit diagonal (dashed gray), and the weighted detection function d_k from Equation A6 (gray, right axis). The detected knee point (green) marks the transition from bare stem to crown.

 doi:10.20944/preprints202604.1825.v1

References

1. Marcozzi, A.; Wells, L.; Parsons, R.; Mueller, E.; Linn, R.; Hiers, J.K. FastFuels: Advancing wildland fire modeling with high-resolution 3D fuel data and data assimilation. *Environmental Modelling & Software* **2025**, *183*, 106214. https://doi.org/10.1016/j.envsoft.2024.106214.
2. Tutland, N.J.; Wion, A.P.; "may", C.J.; Hutchings, G.C.; Nowak, H.A.; Gattiker, J.R.; Hiers, J.K.; Linn, R.R.; Pokswinski, S.M.; Margolis, E.Q. Representing 3-dimensional fuels for physics-based fire behavior models: a general framework and case study in a type-converted post-fire shrubfield. *Fire Ecology* **2025**, *21*, 43. https://doi.org/10.1186/s42408-025-00383-2.
3. Gastellu-Etchegorry, J.P.; Yin, T.; Lauret, N.; Cajgfinger, T.; Gregoire, T.; Grau, E.; Feret, J.B.; Lopes, M.; Guilleux, J.; Dedieu, G.; et al. Discrete Anisotropic Radiative Transfer (DART 5) for Modeling Airborne and Satellite Spectroradiometer and LIDAR Acquisitions of Natural and Urban Landscapes. *Remote Sensing* **2015**, *7*, 1667–1701. https://doi.org/10.3390/rs70201667.
4. Winiwarter, L.; Esmorís Pena, A.M.; Weiser, H.; Anders, K.; Martínez Sánchez, J.; Searle, M.; Höfle, B. Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning. *Remote Sensing of Environment* **2022**, *269*, 112772. https://doi.org/10.1016/j.rse.2021.112772.
5. Xiao, S.; Fei, S.; Li, Q.; Zhang, B.; Chen, H.; Xu, D.; Cai, Z.; Bi, K.; Guo, Y.; Li, B.; et al. The Importance of Using Realistic 3D Canopy Models to Calculate Light Interception in the Field. *Plant Phenomics* **2023**, *5*, 0082. https://doi.org/10.34133/plantphenomics.0082.
6. Bryson, M.; Wang, F.; Allworth, J. Using Synthetic Tree Data in Deep Learning-Based Tree Segmentation Using LiDAR Point Clouds. *Remote Sensing* **2023**, *15*, 2380. Number: 9, https://doi.org/10.3390/rs15092380.
7. Raverta Capua, F.; Schandin, J.; De Cristóforis, P. Training Point-Based Deep Learning Networks for Forest Segmentation with Synthetic Data. In Proceedings of the Pattern Recognition: 27th International Conference, ICPR 2024, Kolkata, India, December 1–5, 2024, Proceedings, Part IV, Berlin, Heidelberg, dec 2024; pp. 64–80. https://doi.org/10.1007/978-3-031-78128-5_5.
8. Buonocore, L.; Yates, J.; Valentini, R. A Proposal for a Forest Digital Twin Framework and Its Perspectives. *Forests* **2022**, *13*, 498. https://doi.org/10.3390/f13040498.
9. Reinhardt, E.; Scott, J.; Gray, K.; Keane, R. Estimating canopy fuel characteristics in five conifer stands in the western United States using tree and stand measurements. *Canadian Journal of Forest Research* **2006**, *36*, 2803–2814. https://doi.org/10.1139/X06-157.
10. Scott, J.H.; Reinhardt, E.D. Assessing crown fire potential by linking models of surface and crown fire behavior. *Res. Pap. RMRS-RP-29. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 59 p.* **2001**, *29*. https://doi.org/10.2737/RMRS-RP-29.
11. Ferrarese, J.; Affleck, D.; Seielstad, C. Conifer crown profile models from terrestrial laser scanning. *Silva Fennica* **2015**, *49*. https://doi.org/10.14214/sf.1106.
12. Purves, D.W.; Lichstein, J.W.; Pacala, S.W. Crown Plasticity and Competition for Canopy Space: A New Spatially Implicit Model Parameterized for 250 North American Tree Species. *PLOS ONE* **2007**, *2*, e870. https://doi.org/10.1371/journal.pone.0000870.
13. Seidl, R.; Rammer, W.; Scheller, R.M.; Spies, T.A. An individual-based process model to simulate landscape-scale forest ecosystem dynamics. *Ecological Modelling* **2012**, *231*, 87–100. https://doi.org/10.1016/j.ecolmodel.2012.02.015.
14. Banerjee, T. The Role of Canopy Turbulence in Wildland Fire Behavior. *Annual Review of Fluid Mechanics* **2026**, *58*, 473–502. https://doi.org/10.1146/annurev-fluid-112723-062216.
15. Linn, R.R.; Hiers, J.K.; O'Brien, J.J.; Yedinak, K.; Hoffman, C.; Canfield, J.; Robinson, D.; Goodrick, S. Wildland fire entrainment: The missing link between wildland fire and its environment. *PNAS Nexus* **2025**, *4*, pgae576. https://doi.org/10.1093/pnasnexus/pgae576.
16. Loudermilk, E.L.; O'Brien, J.J.; Goodrick, S.L.; Linn, R.R.; Skowronski, N.S.; Hiers, J.K. Vegetation's influence on fire behavior goes beyond just being fuel. *Fire Ecology* **2022**, *18*, 9. https://doi.org/10.1186/s42408-022-00132-9.
17. Marcozzi, A.A.; Johnson, J.V.; Parsons, R.A.; Flanary, S.J.; Seielstad, C.A.; Downs, J.Z. Application of LiDAR Derived Fuel Cells to Wildfire Modeling at Laboratory Scale. *Fire* **2023**, *6*, 394. Number: 10, https://doi.org/10.3390/fire6100394.
18. Jeronimo, S.M.A.; Kane, V.R.; Churchill, D.J.; McGaughey, R.J.; Franklin, J.F. Applying LiDAR Individual Tree Detection to Management of Structurally Diverse Forest Landscapes. *Journal of Forestry* **2018**, *116*, 336–346. https://doi.org/10.1093/jofore/fvy023.

 doi:10.20944/preprints202604.1825.v1

19. Creasy, M.B.; Tinkham, W.T.; Hoffman, C.M.; Vogeler, J.C. Potential for individual tree monitoring in ponderosa pine dominated forests using unmanned aerial system structure from motion point clouds. *Canadian Journal of Forest Research* **2021**, *51*, 1093–1105. https://doi.org/10.1139/cjfr-2020-0433.
20. Tinkham, W.T.; Swayze, N.C.; Hoffman, C.M.; Lad, L.E.; Battaglia, M.A. Modeling the Missing DBHs: Influence of Model Form on UAV DBH Characterization. *Forests* **2022**, *13*, 2077. Number: 12, https://doi.org/10.3390/f13122077.
21. Morford, S.L.; Allred, B.W.; Coons, S.P.; Marcozzi, A.A.; McCord, S.E.; Smith, J.T.; Naugle, D.E. A 0.6-meter resolution canopy height and structure model for the contiguous United States, 2025. ISSN: 2692-8205 Pages: 2025.12.12.694075 Section: New Results, https://doi.org/10.64898/2025.12.12.694075.
22. Murphy, M.C.; Loudermilk, E.L.; Pokswinski, S.; Williams, B.; Link, E.; Lienesch, L.; Douglas, L.; Skowronski, N.; Gallagher, M.; Maxwell, A.; et al. Terrestrial 3D Laser Scanning for Ecosystem and Fire Effects Monitoring, 2024. Pages: 2024.04.09.587551 Section: Confirmatory Results, https://doi.org/10.1101/2024.04.09.587551.
23. Roten, D.; Wells, L.; Crawl, D.; Parsons, R.A.; Marcozzi, A.; Linn, R.R.; Hiers, K.; Altintas, I. TrueTrees: A Scalable Workflow for the Integration of Airborne LiDAR Scanning Data into Fuel Models for Prescribed Fire Simulations. In Proceedings of the 2023 IEEE 19th International Conference on e-Science (e-Science), oct 2023, pp. 1–10. https://doi.org/10.1109/e-Science58273.2023.10254923.
24. Schäfer, J.; Weiser, H.; Winiwarter, L.; Höfle, B.; Schmidtlein, S.; Fassnacht, F.E. Generating synthetic laser scanning data of forests by combining forest inventory information, a tree point cloud database and an open-source laser scanning simulator. *Forestry: An International Journal of Forest Research* **2023**, *96*, 653–671. https://doi.org/10.1093/forestry/cpad006.
25. Li, W.; Hu, X.; Su, Y.; Tao, S.; Ma, Q.; Guo, Q. A new method for voxel-based modelling of three-dimensional forest scenes with integration of terrestrial and airborne LiDAR data. *Methods in Ecology and Evolution* **2024**, *15*, 569–582. _eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.14290, https://doi.org/10.1111/2041-210X.14290.
26. Puliti, S.; Lines, E.R.; Müllerová, J.; Frey, J.; Schindler, Z.; Straker, A.; Allen, M.J.; Winiwarter, L.; Rehush, N.; Hristova, H.; et al. Benchmarking tree species classification from proximally sensed laser scanning data: Introducing the FOR-species20K dataset. *Methods in Ecology and Evolution* **2025**, *16*, 801–818. _eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.14503, https://doi.org/10.1111/2041-210X.14503.
27. Lipman, Y.; Chen, R.T.Q.; Ben-Hamu, H.; Nickel, M.; Le, M. Flow Matching for Generative Modeling, 2023. arXiv:2210.02747 [cs], https://doi.org/10.48550/arXiv.2210.02747.
28. Liu, X.; Gong, C.; Liu, Q. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, 2022. arXiv:2209.03003 [cs], https://doi.org/10.48550/arXiv.2209.03003.
29. Luo, S.; Hu, W. Diffusion Probabilistic Models for 3D Point Cloud Generation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), jun 2021, pp. 2836–2844. ISSN: 2575-7075, https://doi.org/10.1109/CVPR46437.2021.00286.
30. Zeng, X.; Vahdat, A.; Williams, F.; Gojcic, Z.; Litany, O.; Fidler, S.; Kreis, K. LION: Latent Point Diffusion Models for 3D Shape Generation, 2022. arXiv:2210.06978 [cs], https://doi.org/10.48550/arXiv.2210.06978.
31. Lee, J.J.; Li, B.; Benes, B. Latent L-systems: Transformer-based Tree Generator. *ACM Trans. Graph.* **2023**, *43*, 7:1–7:16. https://doi.org/10.1145/3627101.
32. Zhou, X.; Li, B.; Benes, B.; Fei, S.; Pirk, S. DeepTree: Modeling Trees with Situated Latents, 2023. arXiv:2305.05153 [cs], https://doi.org/10.48550/arXiv.2305.05153.
33. Wang, H.; Zhang, B.; Klein, J.; Michels, D.L.; Yan, D.; Wonka, P. Autoregressive Generation of Static and Growing Trees, 2025. arXiv:2502.04762 [cs], https://doi.org/10.48550/arXiv.2502.04762.
34. Lee, J.J.; Li, B.; Beery, S.; Huang, J.; Fei, S.; Yeh, R.A.; Benes, B. Tree-D Fusion: Simulation-Ready Tree Dataset from Single Images with Diffusion Priors, 2024. arXiv:2407.10330 [cs] version: 1, https://doi.org/10.48550/arXiv.2407.10330.
35. Xu, H.; Huai, Y.; Nie, X.; Meng, Q.; Zhao, X.; Pei, X.; Lu, H. Diff-Tree: A Diffusion Model for Diversified Tree Point Cloud Generation with High Realism. *Remote Sensing* **2025**, *17*, 923. https://doi.org/10.3390/rs17050923.
36. Stone, G.; Sarker, S.; Greenberg, J.; Tavakkoli, A. Generating Synthetic Tree Point Clouds for Deep Learning Applications in Remote Sensing. In Proceedings of the Advances in Visual Computing; Bebis, G.; Patel, V.; Gu, J.; Panetta, J.; Gingold, Y.; Johnsen, K.; Arefin, M.S.; Dutta, S.; Biswas, A., Eds., Cham, 2025; pp. 3–14. https://doi.org/10.1007/978-3-031-77389-1_1.

 doi:10.20944/preprints202604.1825.v1

37. Bornand, A.; Abegg, M.; Morsdorf, F.; Rehush, N. Completing 3D point clouds of individual trees using deep learning. *Methods in Ecology and Evolution* **2024**, *15*, 2010–2023. _eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.14412, https://doi.org/10.1111/2041-210X.14412.
38. Castorena, J.; Loudermilk, E.L.; Pokswinski, S.; Linn, R. From canopy to ground via ForestGen3D: Learning cross-domain generation of 3D forest structure from aerial-to-terrestrial LiDAR. *ISPRS Journal of Photogrammetry and Remote Sensing* **2026**, *235*, 363–382. https://doi.org/10.1016/j.isprsjprs.2026.03.009.
39. Hartsook, T.; Tavakkoli, A.; Greenberg, J. A Conditional Variational Autoencoder to Learn Mappings Between ALS and TLS Measured Forests. In Proceedings of the Advances in Visual Computing; Bebis, G.; Ye, J.; Wang, Y.; Konaković Luković, M.; Kalantari, N.K.; Cho, I.; Yang, Y.; Dimara, E.; Brehmer, M., Eds., Cham, 2026; pp. 261–271. https://doi.org/10.1007/978-3-032-14495-9_20.
40. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need, 2023. arXiv:1706.03762 [cs], https://doi.org/10.48550/arXiv.1706.03762.
41. Bao, F.; Nie, S.; Xue, K.; Cao, Y.; Li, C.; Su, H.; Zhu, j. All are Worth Words: A ViT Backbone for Diffusion Models, 2023. arXiv:2209.12152 [cs], https://doi.org/10.48550/arXiv.2209.12152.
42. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, 2020. arXiv:2003.08934 [cs], https://doi.org/10.48550/arXiv.2003.08934.
43. Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis, 2024. arXiv:2403.03206 [cs], https://doi.org/10.48550/arXiv.2403.03206.
44. Lipman, Y.; Havasi, M.; Holderrieth, P.; Shaul, N.; Le, M.; Karrer, B.; Chen, R.T.Q.; Lopez-Paz, D.; Ben-Hamu, H.; Gat, I. Flow Matching Guide and Code, 2024, [arXiv:cs.LG/2412.06264].
45. Dormand, J.R.; Prince, P.J. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics* **1980**, *6*, 19–26. https://doi.org/10.1016/0771-050X(80)90013-3.
46. Ho, J.; Salimans, T. Classifier-Free Diffusion Guidance, 2022. arXiv:2207.12598 [cs], https://doi.org/10.48550/arXiv.2207.12598.
47. Pokswinski, S.; Gallagher, M.R.; Skowronski, N.S.; Loudermilk, E.L.; Hawley, C.; Wallace, D.; Everland, A.; Wallace, J.; Hiers, J.K. A simplified and affordable approach to forest monitoring using single terrestrial laser scans and transect sampling. *MethodsX* **2021**, *8*, 101484. https://doi.org/10.1016/j.mex.2021.101484.
48. Wang, P.S. OctFormer: Octree-based Transformers for 3D Point Clouds. *ACM Trans. Graph.* **2023**, *42*, 155:1–155:11. https://doi.org/10.1145/3592131.
49. Wu, X.; Jiang, L.; Wang, P.S.; Liu, Z.; Liu, X.; Qiao, Y.; Ouyang, W.; He, T.; Zhao, H. Point Transformer V3: Simpler, Faster, Stronger, 2024. arXiv:2312.10035 [cs], https://doi.org/10.48550/arXiv.2312.10035.
50. Jaegle, A.; Gimeno, F.; Brock, A.; Zisserman, A.; Vinyals, O.; Carreira, J. Perceiver: General Perception with Iterative Attention, 2021. arXiv:2103.03206, https://doi.org/10.48550/arXiv.2103.03206.
51. Dao, Q.; Phung, H.; Nguyen, B.; Tran, A. Flow Matching in Latent Space, 2023. arXiv:2307.08698 [cs], https://doi.org/10.48550/arXiv.2307.08698.
52. Laino, D.; Cabo, C.; Ordóñez, C.; Bolanos, R.; Janvier, R.; Giulioni, F.; Herrmann, M.; Hudak, A.; Parsons, R.; Santin, C. SegmentedForests: a labelled dataset of terrestrial LiDAR point clouds for semantic segmentation of forests. *Forestry: An International Journal of Forest Research* **2026**, *99*, cpaf062. https://doi.org/10.1093/forestry/cpaf062.
53. Tenny, J.T.; Sankey, T.T.; Munson, S.M.; Sánchez Meador, A.J.; Goetz, S.J. Canopy and surface fuels measurement using terrestrial lidar single-scan approach in the Mogollon Highlands of Arizona. *International Journal of Wildland Fire* **2025**, *34*, WF24221. https://doi.org/10.1071/WF24221.
54. Döllner, J.; de Amicis, R.; Burmeister, J.M.; Richter, R. Forests in the Digital Age: Concepts and Technologies for Designing and Deploying Forest Digital Twins. In Proceedings of the Proceedings of the 28th International ACM Conference on 3D Web Technology, New York, NY, USA, oct 2023; Web3D '23, pp. 1–12. https://doi.org/10.1145/3611314.3616067.

 doi:10.20944/preprints202604.1825.v1

55. Calders, K.; Newnham, G.; Burt, A.; Murphy, S.; Raumonen, P.; Herold, M.; Culvenor, D.; Avitabile, V.; Disney, M.; Armston, J.; et al. Nondestructive estimates of above-ground biomass using terrestrial laser scanning. *Methods in Ecology and Evolution* **2015**, *6*, 198–208. _eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.12301, https://doi.org/10.1111/2041-210X.12301.
56. Satopaa, V.; Albrecht, J.; Irwin, D.; Raghavan, B. Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. In Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops, jun 2011, pp. 166–171. ISSN: 2332-5666, https://doi.org/10.1109/ICDCSW.2011.20.