A Behavioral Analysis of Machine Learning for Intrinsic 3D Tree Branching Structures

M. Charitha Lakshan, K. Damitha Sandaruwan, L. N. C. De Silva
University of Colombo, School of Computing, Colombo 00700, Sri Lanka

***Abstract*—Modeling realistic tree structures remains a challenging problem in computer graphics due to the complex interaction between intrinsic growth patterns and environmental influences. Traditional procedural methods provide strong control over environmental adaptation, while learning-based approaches aim to capture branching behavior directly from data. However, learning-based methods typically entangle intrinsic and environmental factors within a single representation, and are trained predominantly on procedurally generated trees, leaving their behavior under real-world structural noise largely uncharacterized. This study isolates the first half of this problem and asks how well machine learning models learn intrinsic 3D branching structure, hierarchy, geometry, and local topology, independently of environmental influence. Trees are represented as node graphs with intrinsic attributes, and local branching prediction is decomposed into a three-class topology classifier and two cascaded geometry regressors. Across controlled experiments on synthetic and LiDAR-reconstructed datasets, models trained on synthetic trees learn branching topology reliably (98.75% test accuracy), while performance degrades substantially on noisy real-world skeletons (50.29%), revealing the sensitivity of intrinsic-structure learning to data quality. Feature ablation shows that structural attributes, rather than spatial position or orientation, drive most of the predictive performance, raising classification accuracy from 55.55% to 97.80%. A perceptual study (25 participants, 625 pairwise trials) shows that generated trees are selected as more realistic in 33.4% of comparisons, a non-trivial proportion, though reference trees remain significantly preferred. Together, these findings characterize both the promise and the limits of purely data-driven intrinsic branching models, and motivate a hybrid pipeline in which learned intrinsic growth is combined with explicit environmental modeling.**

***Keywords*—Tree modeling; machine learning; intrinsic branching; feature representation; 3D tree reconstruction; synthetic data; behavioral analysis; hybrid modeling**

I. Introduction

Trees are essential elements in virtual environments such as games, simulations, and interactive 3D scenes. In these applications, vegetation modeling prioritizes visual and structural plausibility over strict biological realism. Tree models should appear natural and avoid repetitive or artificial branching patterns that reduce immersion. Unlike botanical studies, which require accurate growth mechanisms, many virtual environments only demand perceptually convincing structures. Consequently, many approaches focus on intrinsic branching properties, such as hierarchy, variation, and proportional relationships, rather than exact biological processes.

Recent data-driven methods, including DeepTree-style [1] approaches, have demonstrated that local branching behavior can be learned using machine learning. These methods typically formulate tree growth as a local prediction problem, where branching structure and geometry are inferred from node-level features. However, learning from real-world scanned trees remains challenging due to noise, missing branches, and reconstruction artifacts, which reduce structural consistency and learning accuracy.

In this work, we present a behavioral analysis of machine learning models trained to learn intrinsic 3D tree branching patterns without modeling environmental effects. Rather than reproducing biologically accurate growth, we focus on generating visually plausible branching structures for virtual environments. As illustrated in Fig. 1, tree growth is modeled as an iterative local prediction process based on intrinsic node attributes. A classifier predicts branching decisions, while conditional regression models estimate branch geometry, enabling the generation of structurally consistent trees. We further compare models trained on synthetic and real scanned datasets to analyze the impact of data quality on learning performance. Results show that intrinsic branching patterns can be learned reliably under controlled conditions, whereas noisy real-world data introduces significant limitations. These findings support a decoupled modeling strategy in which intrinsic growth is learned independently from environmental adaptation handled by explicit procedural or mathematical models. Accordingly, this work focuses on analyzing the behavior of machine learning models when learning intrinsic branching structures, rather than proposing a novel machine learning architecture, providing a foundation for future hybrid tree modeling pipelines.

II. Related Work

Tree modeling has been widely studied in computer graphics, with approaches ranging from procedural rule-based systems to data-driven and learning-based methods. A central challenge in this area is modeling both intrinsic branching

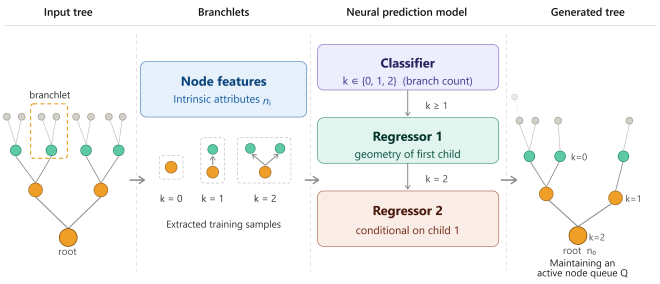


Fig. 1. Overview of the learning framework for behavioral analysis of intrinsic 3D tree branching. Branchlets are represented by intrinsic node attributes n_i. A classifier predicts the number of child branches ($k \in \{0, 1, 2\}$), followed by conditional regression models estimating branch geometry.

structure and environmental adaptation, including responses to light, obstacles, and neighboring vegetation. Procedural methods have long been used for this purpose due to their interpretability, controllability, and computational efficiency. One of the most influential approaches is the L-system framework [2], which represents plant growth using symbolic rewriting rules and has been extended to incorporate phototropism and space competition. Other procedural models explicitly simulate environmental influence through light-driven growth and spatial optimization. Beneš [3] introduced phototropic plant development models based on light availability, while the space colonization algorithm of Runions et al. [4] modeled branching as an iterative attraction process guided by available space. More biologically inspired approaches, such as the self-organizing tree model of Pałubicki et al. [5], simulate local competition for light and space to produce complex emergent tree structures. In all of these formulations, however, both intrinsic growth and environmental response are expressed as hand-authored rules, so structural realism depends on manual parameter tuning rather than on branching behavior observed in real trees.

Recent procedural work has focused on interactive and adaptive growth. Plastic Trees [6] introduced skeleton-based models that dynamically adjust branch orientation in response to light, obstacles, and nearby vegetation, while virtual ecosystem approaches enable plants to interact with surrounding objects and optimize spatial occupation [7]. GPU-accelerated methods incorporating ray-traced light simulation have further improved environmental realism and computational performance [8]. Inverse procedural approaches have also been proposed to estimate procedural parameters directly from existing tree geometry, bridging rule-based modeling and real-world observations [9]. These methods provide strong control over environmental adaptation through explicit parameterized rules, making them well suited for environment-driven modeling in virtual scenes. Their expressiveness, however, remains bounded by the rule set chosen in advance, and intrinsic branching is still specified rather than inferred from data.

More recently, machine learning and hybrid approaches have explored learning branching behavior directly from data. DeepTree [1] formulates tree growth as a local prediction problem, where branching decisions and geometric attributes are inferred from node-level spatial context. Transformer-based autoregressive models have further demonstrated the ability to generate hierarchical tree structures and temporal growth sequences directly from learned representations [10]. Hybrid methods such as Latent L-systems [11] combine neural learning with grammar-based representations, allowing branching rules to be learned implicitly while retaining procedural structure, whereas TreeStructor [12] targets faithful reconstruction of scanned geometry rather than the generation of novel structures. Two properties of this body of work are relevant here. First, these models are predominantly trained on procedurally generated data, a choice their authors attribute to the limited availability of large collections of reconstructed real trees [1]; consequently, how such models behave when trained on noisy reconstructed skeletons remains largely uncharacterized. Second, intrinsic structure and environmental context are typically encoded jointly, whether within a single node representation or a learned sequence, so the contribution of intrinsic attributes alone cannot readily be isolated or analyzed. These observations motivate a systematic behavioral analysis of how machine learning models learn intrinsic 3D tree branching structure independently of environmental influences, and whether such learning can serve as a reliable foundation for a broader decoupled modeling strategy.

III. Problem Formulation and Representation

In this work, a tree is represented as a rooted graph $\mathcal{T}$, where nodes correspond to branching points or segment endpoints, and edges represent parent-child relationships between connected branches. This graph-based representation is widely used in both procedural and data-driven tree modeling, as it provides a compact and flexible description of hierarchical structure and local geometry.

Each node in the tree graph is represented as a tuple of intrinsic attributes that describe its local geometric and structural properties. Following the representation style introduced in DeepTree [1], nodes are modeled as attribute tuples rather than purely topological elements, enabling local branching behavior to be learned from node-level context. In contrast to DeepTree, the attribute set used in this work is intentionally simplified. Only attributes that are consistently available across both real scanned trees and synthetically generated data are included. Attributes related to biological age, species, or other global physiological properties are omitted, as they are either unavailable or unreliable in the considered datasets.

Formally, each node $n_i \in \mathcal{T}$ is defined as:

$$n_i = \left(\mathbf{p}_i, \mathbf{q}_i, r_i, r^{root}, o_i, d_i, d_i^{root}, d_i^{branch}\right) \tag{1}$$

where, $\mathbf{p}_i \in \mathbb{R}^3$ denotes the 3D position of the node, $\mathbf{q}_i \in \mathbb{R}^4$ is a unit quaternion representing the local branch orientation with respect to its parent node, $r_i \in \mathbb{R}^+$ is the local branch radius, and $r^{root} \in \mathbb{R}^+$ denotes the radius of the root node, which is shared by all nodes in the tree and provides global scale context. The value $o_i \in \mathbb{N}$ denotes the branch order. The value $d_i \in \mathbb{R}$ represents the distance to the parent node. In addition, $d_i^{root} \in \mathbb{R}$ denotes the cumulative distance from the root node, and $d_i^{branch} \in \mathbb{R}$ represents the distance from the start of the current branch. All attributes describe intrinsic structural properties and do not encode any explicit environmental information.

Tree growth is modeled as a local prediction problem. Instead of predicting the full tree structure at once, growth behavior is inferred at the node level using local information. For each node, a local growth neighborhood-referred to as a "branchlet" (Fig. 2) is defined, consisting of a parent node and its immediate children together with their connecting edges. Based on this local representation, the learning task is divided into two components: a classification task that predicts the number of child branches produced by a node, and regression tasks that predict geometric properties of each child branch, such as relative orientation, length, and thickness. Together, these predictions describe the intrinsic branching structure in a local and incremental manner.

A key assumption of this formulation is that environmental factors are not considered. External influences such as sunlight, wind, gravity, terrain constraints, and interactions with surrounding vegetation are intentionally excluded from both the

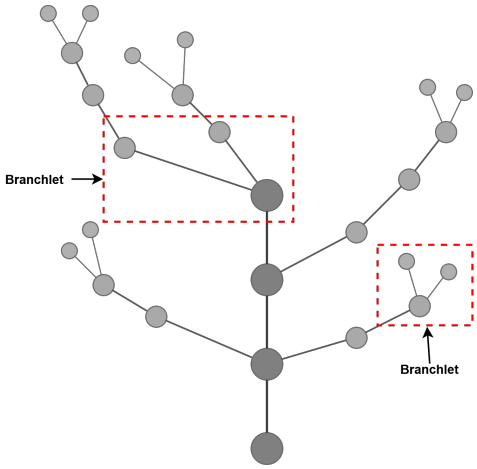


Fig. 2. Local branchlet representation used in intrinsic branching prediction.

representation and the learning process. The objective of this work is therefore not to model adaptive growth behavior, but to analyze how the proposed learning framework behaves when trained exclusively on intrinsic branching patterns, without exposure to environmental influences. This formulation supports a modular modeling approach, in which intrinsic growth prediction can later be combined with a separate environmental adaptation stage. The following section describes how this local prediction problem is realized through a classification and regression pipeline operating on the intrinsic node representation defined above.

IV. Learning Intrinsic Branching

This section describes how intrinsic tree branching behavior is learned using the node representation introduced earlier. The learning pipeline consists of two main stages: branch structure prediction and branch geometry prediction. First, a classification model predicts the number of child branches emerging from a parent node in a local context. Based on this prediction, regression models are then used to estimate the geometric properties of each child branch. The resulting predictions are applied by a tree generation procedure to incrementally construct branching structures.

A. Branch Structure Prediction

The first learning task focuses on predicting local branching topology. Given the intrinsic attributes of a parent node, a classification model predicts the number of child branches produced by that node. The classification output is restricted to three classes: 0, 1, or 2 children.

This restriction is motivated by the distribution of branching patterns observed in both real scanned trees and synthetically generated trees. During dataset analysis, nodes with three or more child branches were found to be rare, accounting for approximately 3% or less of all branching events. Including these cases introduced strong class imbalance and negatively affected training stability. Therefore, nodes with more than two children are excluded from training, allowing the classifier to focus on the most common and structurally relevant branching patterns.

A lightweight feed-forward neural network is used for branch count classification (Fig. 3). In contrast to more complex classifier designs used in prior work such as DeepTree, the proposed architecture is intentionally simplified to improve robustness and interpretability under noisy real-world data conditions. The model takes the intrinsic attributes of the parent node as input and outputs class logits corresponding to the three child-count categories. The network is trained using the cross-entropy loss function, which encourages accurate prediction of the discrete branching classes. This design emphasizes reliable local branching prediction rather than modeling global tree structure.

B. Branch Geometry Prediction

Once the number of child nodes is determined, the next task is to predict the geometric properties of each child node. These properties include relative orientation with respect to the parent, as well as geometric attributes such as edge length and node radius.

To model this, two regression networks are employed. The first regressor predicts the geometry of the first child node. The second regressor predicts the geometry of the second child node when two children are present, and operates in a conditional manner based on the first child. Specifically, during tree generation, the second regressor takes as input the intrinsic attributes of the parent node together with the predicted attributes of the first child, forming a cascaded prediction structure. During training, however, the second regressor is conditioned on the ground-truth attributes of the first child rather than the predicted outputs of the first regressor. This design avoids error propagation during training and allows the model to learn stable conditional geometric relationships.

The regression architecture is influenced by the design used in DeepTree, with a multi-head structure that separates orientation prediction from geometric attribute estimation (Fig. 4). Each regressor consists of two prediction heads: one for orientation, represented as a quaternion, and another for geometric attributes, including relative edge length and node radius. The models take intrinsic node attributes as input and output continuous predictions for these properties.

The regression networks are trained using a combination of loss functions. Orientation prediction is supervised using a cosine similarity loss between the predicted and ground-truth orientations, while geometric attributes are optimized using mean squared error (MSE). These losses are jointly minimized to ensure accurate prediction of local branching geometry.

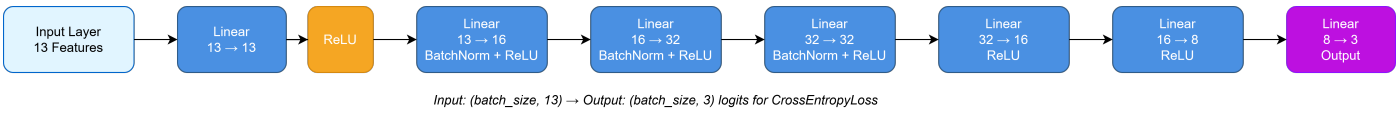


Fig. 3. Architecture of the child-count classification network used to predict the number of child branches (0, 1, or 2) from intrinsic node attributes.

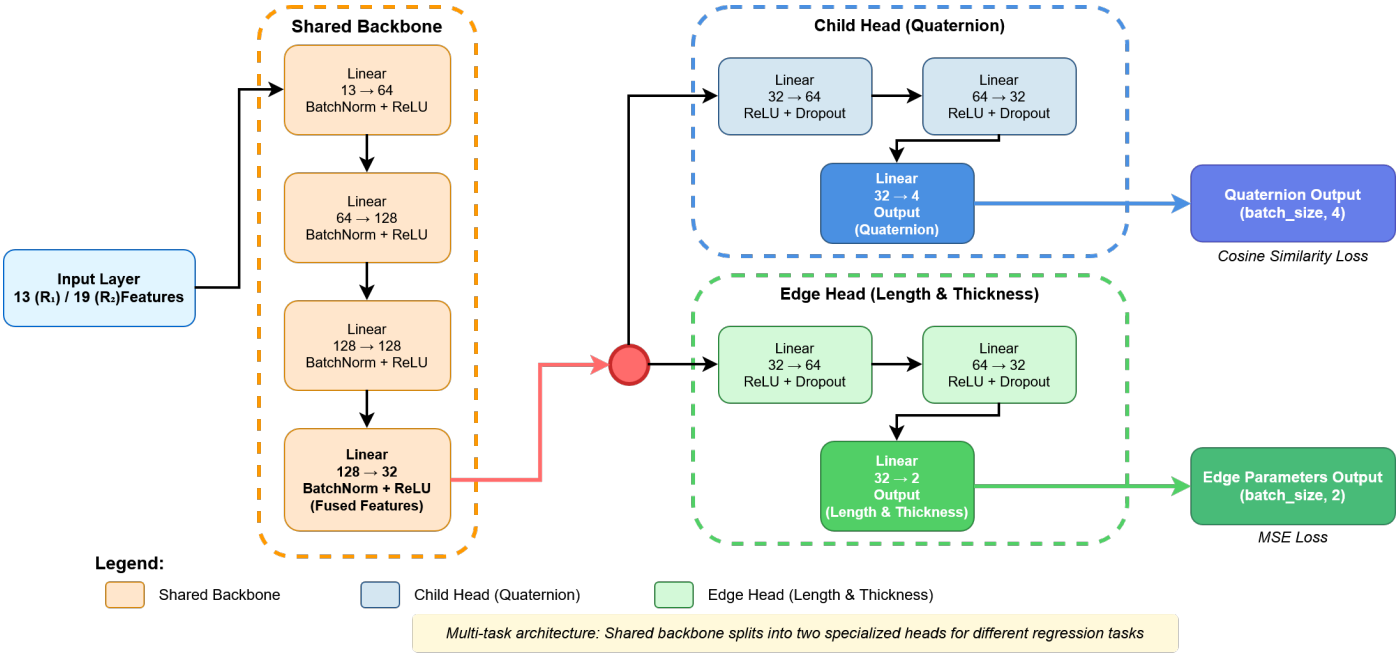


Fig. 4. Multi-task regression architecture used for branch geometry prediction.

C. Incremental Tree Generation

The trained classification and regression models are used to generate tree structures in an incremental manner. Tree generation begins from a manually initialized root node, whose intrinsic attributes are specified explicitly. These include the root position, orientation, branch radius, and initial distance values. Manual initialization of the root node is required, as the learning models operate locally and require an initial node context to initiate growth.

Starting from the root node, tree construction proceeds by iteratively expanding active nodes, as summarized in Algorithm in the Fig. 5. For each active node, the child-count classifier predicts whether the node produces zero, one, or two child branches. For nodes that produce children, the corresponding regression models predict geometric properties of each child branch, including relative orientation, edge length, and branch thickness. Child node positions and orientations are computed by applying the predicted transformations to the parent node.

Predicted branching topology alone does not guarantee termination: prior work reports that locally predicted child counts may fail to produce terminal nodes for some configurations, allowing generation to continue indefinitely [1]. To avoid uncontrolled growth and to improve stability, an existence probability is associated with each node. This probability is propagated from parent to child and gradually reduced as the depth of the tree increases. Child nodes are only created if their sampled existence probability exceeds a random threshold. Newly generated child nodes are added to a queue of active nodes and processed in subsequent iterations. The generation process terminates when no active nodes remain or when a predefined node limit is reached. This procedure models intrinsic branching behavior only and does not include any environmental constraints.

Fig. 6 presents representative examples of tree structures generated using the incremental generation procedure described above. The generated structures exhibit coherent hierarchical branching organization, with consistent radius tapering across depth levels and plausible bifurcation patterns, indicating that the employed learning framework produces structurally consistent and visually plausible tree forms from purely intrinsic node attributes.

1) Notation and symbols: Let $\mathcal{T}$ denote the generated tree represented as a rooted graph. Each node $n_i \in \mathcal{T}$ is associated with a tuple of intrinsic attributes $\left(\mathbf{p}_i, \mathbf{q}_i, r_i, r^{root}, o_i, d_i, d_i^{root}, d_i^{branch}\right)$, as defined in Section III. The classifier $C(\cdot)$ predicts the number of child branches $k \in \{0, 1, 2\}$ for a given parent node. The regressors $R_1(\cdot)$ and $R_2(\cdot)$ predict geometric properties of the first and second child branches. Each node n_i is additionally associated with an existence probability $p(n_i)$, which controls whether the node is instantiated during generation. The parameter α denotes the probability decay factor applied beyond a depth threshold D_{start}. The queue $\mathcal{Q}$ maintains the set of active nodes, and $N_{\max}$ specifies the maximum number of nodes allowed in the generated tree.

D. Implementation Details

All models were implemented in PyTorch and trained on a single NVIDIA RTX 5060 GPU with an AMD Ryzen 5 7500F CPU and 32 GB of system memory. The intrinsic node tuple of Eq. (1) expands to a 13-dimensional input vector; Regressor 2 additionally receives the four orientation and two geometric attributes of the first child, giving a 19-dimensional input. Layer widths are given in Fig. 3 and Fig. 4, with dropout applied within the regression heads only. Because the cosine-similarity and mean squared error terms of the regression objective differ substantially in numerical scale, the geometric term is weighted so that neither head dominates optimization. No learning rate scheduling, weight decay, or early stopping is applied; the weights obtained at the final epoch are retained. Remaining settings are summarized in Table I.

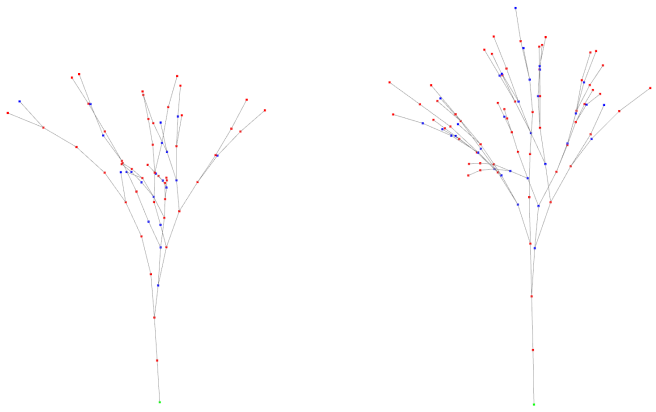
Require: Trained child classifier C
Require: Trained regressors R_1, R_2
Require: Root node n_0 with initial existence probability p_0
Require: Maximum node limit $N_{\max}$
Require: Probability decay factor α
Require: Depth threshold D_{start}
Ensure: Generated tree structure $\mathcal{T}$

Initialize queue $\mathcal{Q} \leftarrow \{n_0\}$
Initialize tree $\mathcal{T} \leftarrow \{n_0\}$
Set $p(n_0) \leftarrow p_0$
while $\mathcal{Q}$ is not empty **and** $|\mathcal{T}| < N_{\max}$ **do**
    Pop parent node n_p from $\mathcal{Q}$
    Predict number of children $k \leftarrow C(n_p)$
    Let d_p be the depth (nodes-to-parent count) of n_p
    if $k \geq 1$ **then**
        Predict child 1 geometry using $R_1(n_p)$
        Compute child depth $d_1 \leftarrow d_p + 1$
        if $d_1 > D_{start}$ **then**
            Apply probability decay $p_1 \leftarrow p(n_p)\,(1-\alpha)$
        else
            $p_1 \leftarrow p(n_p)$
        end if
        if rand() $< p_1$ **then**
            Create child node n_1
            Assign $p(n_1) \leftarrow p_1$
            Add n_1 to $\mathcal{T}$ and $\mathcal{Q}$
        end if
    end if
    if $k = 2$ **then**
        Predict child 2 geometry using $R_2(n_p, R_1(n_p))$
        Compute child depth $d_2 \leftarrow d_p + 1$
        if $d_2 > D_{start}$ **then**
            Apply probability decay $p_2 \leftarrow p(n_p)\,(1-\alpha)$
        else
            $p_2 \leftarrow p(n_p)$
        end if
        if rand() $< p_2$ **then**
            Create child node n_2
            Assign $p(n_2) \leftarrow p_2$
            Add n_2 to $\mathcal{T}$ and $\mathcal{Q}$
        end if
    end if
end while
return $\mathcal{T}$

Fig. 5. Incremental tree generation algorithm.

TABLE I. TRAINING AND GENERATION CONFIGURATION

Parameter	Value
Optimizer	Adam
Learning rate	1×10^{-4}
Batch size	512
Epochs	300
Dropout (regressor heads)	0.2
Geometry loss weight	100
Initial existence probability p_0	1.0
Depth threshold D_{start}	5
Probability decay factor α	0.1
Node limit $N_{\max}$	250
Root candidates sampled	100
Root sampling radius	0.1



(a) Generated tree sample 1 (b) Generated tree sample 2

Fig. 6. Representative examples of generated tree structures.

V. DATASETS

This section describes the datasets used for learning intrinsic branching behavior and the preprocessing steps applied to construct branchlet-level training samples. Two complementary data sources are considered: real-world tree reconstructions derived from LiDAR scans and procedurally generated synthetic tree skeletons. Using both datasets enables analysis of learning behavior under realistic but noisy conditions as well as under controlled structural settings.

A. Real-World Tree Dataset

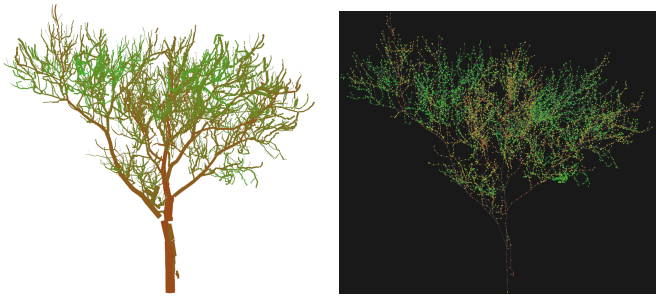
For real-world data, we use the TreeML-Data dataset [13], which contains tree structures reconstructed from terrestrial Light Detection and Ranging (LiDAR) scans using quantitative structure modeling (QSM) (Fig. 7). In this representation, trees are described as graphs of connected cylindrical primitives that approximate branch segments and encode geometric attributes such as position, orientation, radius, and branch order.

Although TreeML-Data does not provide an explicit node-edge skeleton, the cylinder graph preserves hierarchical connectivity and therefore supports derivation of a structural node representation. In this work, starting points of the cylinders are interpreted as nodes while connectivity between segments defines graph edges, enabling conversion to the node-level formulation used for learning.

Because the dataset originates from real measurements, the resulting structures may contain missing branches, uneven sampling, geometric noise, and segmentation inconsistencies. These factors introduce variability at the local branching level and can influence learning behavior, particularly in deeper structural regions. Nevertheless, real-world data remains important for evaluating how intrinsic branching models perform under practical acquisition conditions.

B. Synthetic Tree Dataset

To support controlled evaluation of intrinsic branching behavior, we additionally use synthetic skeletal tree structures released with the CHI24_TreeProject dataset [14] (Fig. 8). These trees are generated using procedural modeling techniques and provided as explicit skeletal graphs describing parent-child relationships and geometric attributes at each node.



(a) QSM cylinder representation (b) Derived node-edge graph

Fig. 7. Real-world tree representation from TreeML-Data.

Compared to LiDAR reconstructions, the synthetic skeletons form clean and fully connected hierarchical structures without scanning artifacts or segmentation noise. Such representations are widely used in procedural vegetation modeling because they expose branching organization directly and allow local growth relationships to be analyzed independently of reconstruction errors.

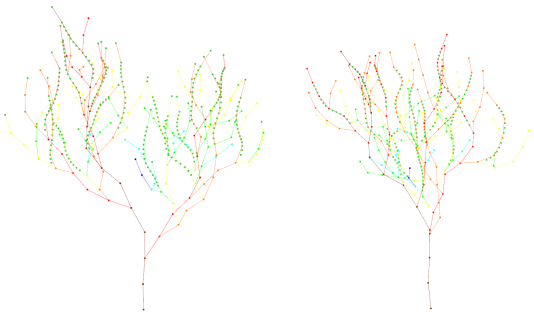
While synthetic trees do not capture the full variability of biological growth, they provide a stable structural environment for studying branching prediction, orientation modeling, and geometric parameter learning. The combination of real reconstructions and synthetic skeletons therefore enables evaluation across both realistic and controlled conditions.

C. Data Preprocessing

Both datasets undergo preprocessing to produce a unified intrinsic node representation suitable for branchlet-level learning. The objective is to extract structural attributes, normalize geometric scale, and construct local parent-child neighborhoods while minimizing implicit environmental bias.

For LiDAR-derived trees, cylindrical segments are converted into node-edge graphs by treating segment centers as nodes while preserving connectivity as hierarchical edges. Intrinsic attributes, including spatial position, node radius, branch order, edge length, cumulative distance from the root, and distance from branch origin, are computed through graph traversal. Local orientation is encoded using quaternion representations derived from parent-child direction vectors.

Because the intrinsic features are derived directly from spatial coordinates, geometric normalization plays an important role in defining the statistical distribution of input features as well as the numerical scale of regression targets. To evaluate its influence, an initial preprocessing study was conducted using synthetic tree structures, where reconstruction artifacts are absent and the effect of normalization can be analyzed in isolation. Two commonly used strategies were considered: uniform (isotropic) scaling, which preserves the original aspect ratios while fitting each tree into a bounded spatial domain, and axis-wise normalization, which independently rescales each spatial dimension to a fixed range. Both strategies enabled accurate learning of branching topology and geometry. However, uniform scaling consistently produced more stable optimization, slightly higher classification accuracy, and lower regression errors, particularly for branch geometry prediction. Based on these observations, uniform scaling was adopted as the default normalization strategy for all subsequent experiments.



(a) (b)

Fig. 8. Synthetic skeletal tree examples from the CHI24_TreeProject dataset.

Synthetic skeletons follow the same structural extraction procedure and additionally undergo controlled rotation augmentation around the vertical axis to improve orientation invariance. Distance measures and orientation encoding are computed consistently with the real-world data.

For each parent node, immediate children are grouped to form branchlets that define classification targets (number of children) and regression targets describing relative child orientation, edge length, and node radius. Despite differences in acquisition and generation processes, preprocessing produces equivalent intrinsic feature representations for both datasets, enabling learning to focus on internal branching structure while maintaining compatibility across data modalities.

The dataset is divided at the tree level to avoid structural overlap between training and evaluation samples. Specifically, 70% of the trees are used for training, 15% for validation, and 15% for testing. All nodes belonging to a given tree are assigned to the same subset, preventing structural information leakage across splits.

VI. Experiments, Results and Evaluation

This section presents a behavioral analysis of the employed learning framework through controlled experiments on intrinsic 3D tree branching structures. The analysis comprises a comparison of learning performance across synthetic and real-world datasets, an intrinsic feature importance study, and a perceptual evaluation of generated tree structures. All quantitative experiments were repeated **five times** using **different random seeds**, with reported metrics representing mean values; figures correspond to the run **closest to the mean**. Prior learning-based tree models are not evaluated under a directly comparable protocol. DeepTree [1] adopts a multi-label formulation in which independent output heads determine whether each potential child is instantiated, and is trained on procedurally generated data under a different geometric normalization, so its reported node-level metrics do not correspond to the three-class formulation and evaluation setting used here. Latent L-systems [11] validates generated structures through aggregate geometric distributions and perceptual metrics rather than node-level prediction error. Moreover, no prior work reports node-level metrics on LiDAR-reconstructed skeletons, the specific data condition examined in Section VI-A. Direct numerical comparison would require re-implementation and retraining under the present protocol, an effort that falls outside the scope of this behavioral analysis. Human perceptual judgment

(Section VI-C) is therefore adopted as the reference point, consistent with prior validation strategies in which generated structures are compared against dataset reference trees.

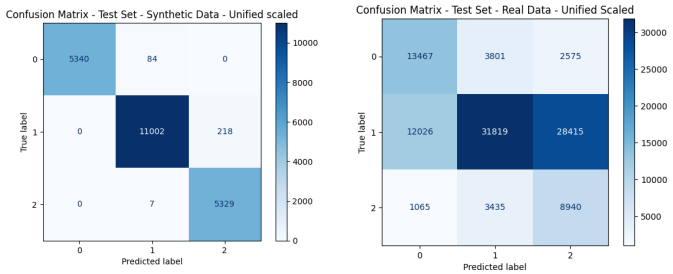
A. Synthetic and Real Data Comparison

We evaluate how the learning model behaves when trained on different data sources, using synthetic and real-world tree data. Classification accuracy, regression error, and convergence behavior are compared under identical conditions.

1) Experimental setup: The dataset split follows the procedure described in Section V, where trees are partitioned at the instance level to prevent structural overlap between training, validation, and test sets. The branching classifier is formulated as a three-class prediction problem corresponding to nodes with zero, one, or two children. To mitigate class imbalance, the training and validation sets are balanced across the three classes using random downsampling, while the test sets retain the original class distributions for unbiased evaluation. This balancing is applied only to the classifier, whereas the regression models are trained on the original data distributions. Uniform scaling, adopted as the default normalization strategy following the preprocessing analysis described in Section V, is applied consistently across all experiments.

2) Branching classification results: The branching classifier was evaluated on both synthetic and real tree datasets to analyze how effectively the intrinsic representation captures branching topology under different conditions. Under identical training settings, both models exhibited stable convergence with no significant signs of overfitting. The model trained on synthetic data converged rapidly and achieved validation accuracy above 98%, whereas the model trained on real tree data converged to a lower but stable accuracy of approximately 60%, indicating that branching prediction is more challenging under real-world structural conditions.

To further analyze prediction behavior, confusion matrices were computed for the held-out test sets. Fig. 9 presents the confusion matrices for both datasets. For the synthetic dataset, the classifier demonstrates strong performance across all three classes, with only a small number of misclassifications between continuation nodes and bifurcation nodes. In contrast, the confusion matrix obtained from the real dataset reveals a larger number of errors between these two classes, while terminal nodes (class 0) remain relatively well identified. This suggests that continuation and bifurcation states often exhibit similar local geometric configurations in real tree skeletons, leading to increased structural ambiguity.



(a) Synthetic dataset. *(b) Real dataset.*

Fig. 9. Confusion matrices for the branching classifier evaluated on the test sets for synthetic and real datasets.

Quantitative results are summarized in Table II. The classifier trained on synthetic data achieves consistently high accuracy across training, validation, and test sets, reflecting the structural consistency of the synthetic trees. In contrast, the model trained on real tree data achieves lower overall accuracy while still maintaining meaningful predictive performance. Because the training and validation sets are class-balanced while the test sets retain the original distribution, raw accuracy on the real dataset understates model performance relative to a majority-class predictor; balanced accuracy across the three classes remains substantially above the 33.3% chance level. This suggests that the intrinsic feature representation remains informative despite the increased geometric variability and reconstruction noise present in real-world tree structures.

3) Regression results: The regression networks predict child branch orientation, represented as a quaternion, together with geometric attributes including branch radius and edge length. The models were trained separately on synthetic and real datasets to analyze how learning behavior varies across different data conditions. Under identical optimization settings, all regression models exhibited stable convergence without significant overfitting. Models trained on synthetic data converged more rapidly and achieved lower final losses, whereas models trained on real data stabilized at higher loss values, reflecting the increased structural variability of real-world tree skeletons.

a) Orientation prediction: Orientation prediction accuracy was evaluated using the angular deviation between the predicted quaternion and the ground-truth branch orientation. Fig. 10 shows the angular error distributions for all regression models. For models trained on synthetic data [Fig. 10(a) and 10(b)], most predictions are concentrated within small angular deviations, indicating high orientation accuracy. Regressor 1 exhibits a tighter concentration of low-angle errors, while Regressor 2 shows a slightly broader distribution, reflecting the increased complexity of predicting secondary branch orientation.

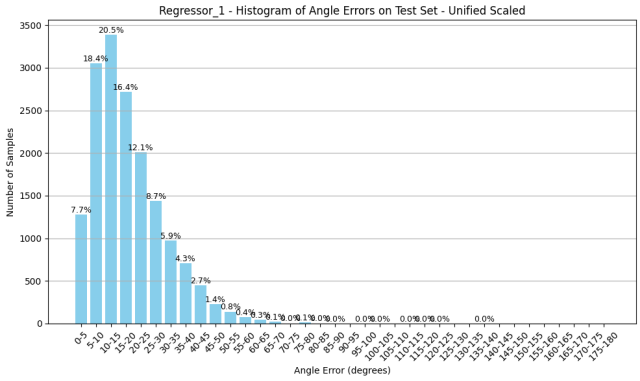
In contrast, models trained on real data [Fig. 10(c) and 10(d)] produce noticeably wider error distributions with longer tails toward larger angular deviations. Although Regressor 1 still maintains a concentration of predictions at lower angles, Regressor 2 shows the widest spread, indicating reduced orientation accuracy under real-world structural conditions. These results suggest that both regressors successfully learn orientation relationships, although prediction accuracy decreases on real data, particularly for secondary branch geometry.

b) Geometric attribute prediction: In addition to orientation, the regression networks estimate branch radius and edge length. Prediction accuracy is evaluated using percentage error relative to ground-truth values. Fig. 11 shows the corresponding error distributions.

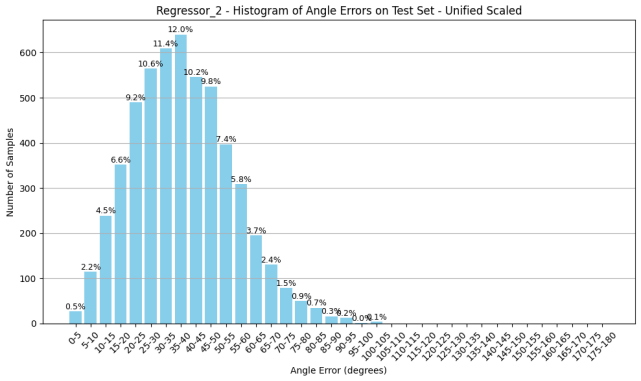
For synthetic data [Fig. 11(a) and 11(b)], both regressors produce highly concentrated error distributions, with most

TABLE II. CLASSIFICATION PERFORMANCE ON SYNTHETIC VS REAL DATA (MEAN ± STANDARD DEVIATION)

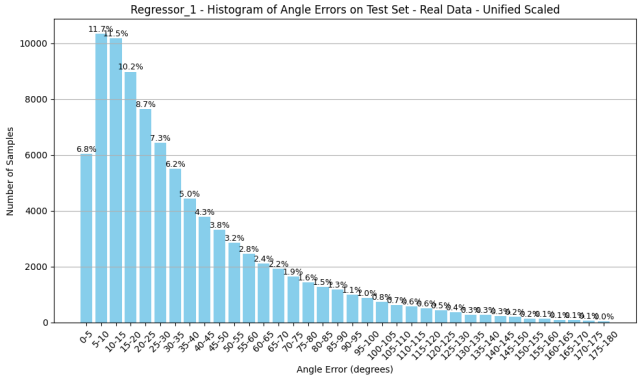
Dataset	Train Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)
Synthetic	98.73 ± 0.08	99.05 ± 0.09	98.75 ± 0.06
Real Trees	59.81 ± 0.06	60.02 ± 0.20	50.29 ± 1.25



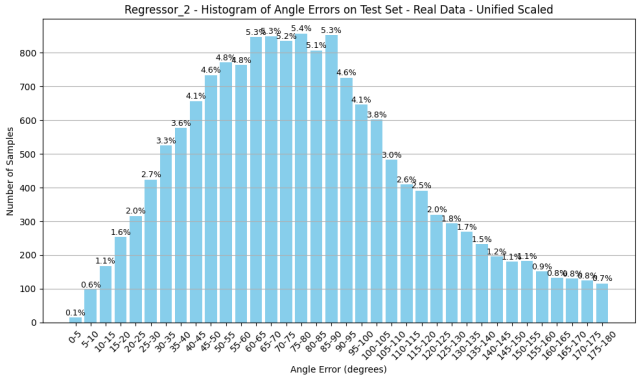
(a) Synthetic data – Regressor 1.



(b) Synthetic data – Regressor 2.

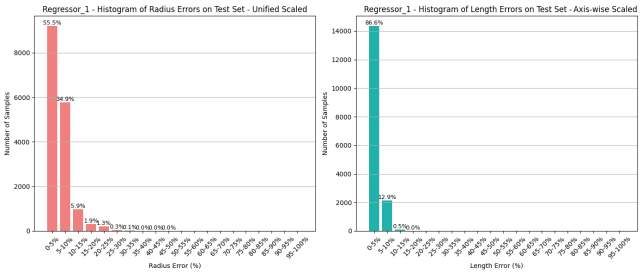


(c) Real data – Regressor 1.

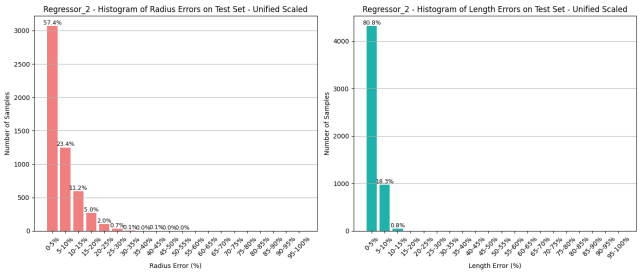


(d) Real data – Regressor 2.

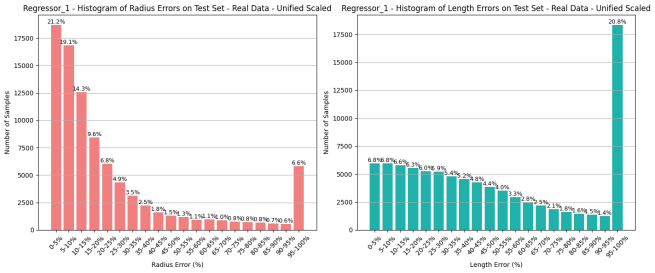
Fig. 10. Angular error distributions for the regression networks trained on synthetic and real datasets.



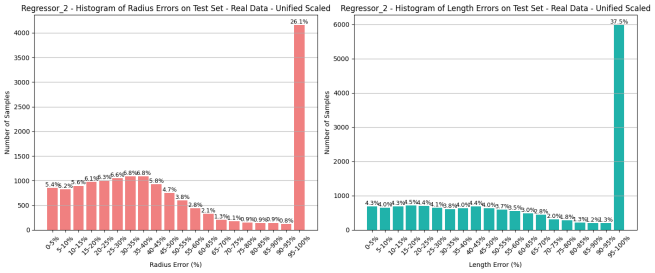
(a) Synthetic data – Regressor 1.



(b) Synthetic data – Regressor 2.



(c) Real data – Regressor 1.



(d) Real data – Regressor 2.

Fig. 11. Radius and edge length prediction error distributions for the regression networks evaluated on the test datasets.

predictions falling within low percentage error ranges. Regressor 1 exhibits slightly tighter distributions than Regressor 2, indicating more accurate and stable geometric predictions for the primary child branch.

In contrast, models trained on real data [Fig. 11(c) and 11(d)] show significantly wider and more irregular error distributions, with long tails extending toward higher error ranges. A noticeable accumulation of samples appears in the highest error bins, indicating occasional large prediction deviations. These effects are more pronounced for Regressor 2, highlighting the increased difficulty of predicting secondary branch geometry in structurally inconsistent real-world trees.

Quantitative results in Table III support these observations, showing substantially higher orientation and geometry losses

TABLE III. REGRESSION PERFORMANCE UNDER SYNTHETIC AND REAL TRAINING DATASETS (MEAN ± STANDARD DEVIATION)

Dataset	Model	Orientation Loss	Geometry Loss
Synthetic Trees	Reg. 1	0.0185 ± 0.0001	$(2.4 \pm 0.55) \times 10^{-6}$
Real Trees	Reg. 1	0.0819 ± 0.0002	$(3.52 \pm 0.08) \times 10^{-5}$
Synthetic Trees	Reg. 2	0.0653 ± 0.0013	$(2.6 \pm 0.55) \times 10^{-6}$
Real Trees	Reg. 2	0.2714 ± 0.0009	$(6.1 \pm 0.00) \times 10^{-5}$

for models trained on real data compared with those trained on synthetic data. Overall, the employed approach learns local geometric relationships reliably under controlled synthetic conditions, while performance degrades on real-world data due to increased structural variability, reconstruction noise, and the presence of more ambiguous branching configurations.

4) Discussion: The results of this experiment reveal a clear difference in learning behavior between models trained on synthetic tree structures and those trained on real-world tree skeleton data. Models trained on synthetic data achieve high classification accuracy and low regression errors, indicating that the intrinsic feature representation effectively captures branching geometry when the underlying structures follow consistent geometric rules. In contrast, models trained on real data exhibit reduced classification accuracy and higher regression errors, although training remains stable and meaningful structural patterns are still learned.

This performance gap can be attributed to the increased structural variability present in real-world tree data. Unlike synthetic trees, which are generated with controlled and consistent geometric relationships, real tree models are reconstructed from scanned data using cylindrical segment approximations. This process introduces irregularities such as noisy branching angles, inconsistent segment lengths, abrupt radius variations, and local reconstruction artifacts. These factors result in ambiguous local geometric configurations, making it more difficult for the model to infer branching topology and accurately predict geometric attributes.

Fig. 12 provides visual examples of these structural characteristics. Compared to synthetic trees, real tree skeletons exhibit geometric inconsistencies that disrupt regular branching patterns. These irregularities contribute to the wider error distributions observed in Fig. 10 and Fig. 11, as well as the reduced classification accuracy shown in Fig. 9 and Table II. Building on these observations, Fig. 13 presents tree structures generated using models trained on real-world data. The trees are produced using the incremental generation process described in Section IV, where branching decisions and geometric attributes are predicted iteratively.

The generated structures exhibit coherent branching behavior at a global level, indicating that the model captures fundamental structural relationships even under challenging data conditions. However, compared to the synthetic data results shown under the section IV, the generated trees display increased variability in local branching patterns, including irregular branch orientations, inconsistent segment lengths, and less uniform structural organization. These effects can be associated with the broader error distributions observed in the regression results, as well as the reduced classification accuracy for branching decisions. Such variability reflects the difficulty of learning consistent structural representations from real-world tree data, where geometric irregularities and reconstruction artifacts introduce ambiguity in local branching configurations. As a result, while the generated trees retain recognizable branching characteristics, their structural consistency is less pronounced than that observed in models trained on synthetic data.

Fig. 12. Real tree examples represented using cylindrical branch segments.

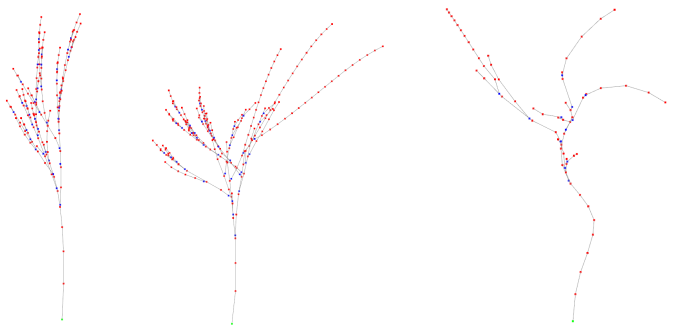


Fig. 13. Tree structures generated using models trained on real-world data. (Only for this, the node limit is set to 1000 for better visualization of the branching patterns.)

Overall, the results demonstrate that the employed framework exhibits consistent behavior when trained on synthetic datasets, while behavioral degradation is observed on real-world data. The reduced performance highlights the influence of noise, geometric inconsistencies, and reconstruction artifacts on both branching decisions and geometric predictions. These findings characterize both the capabilities and the limitations of the learning framework, and suggest that improvements in preprocessing, data acquisition, and geometric representation could further enhance behavioral stability and output quality.

B. Intrinsic Feature Importance Analysis

While the previous analysis examined the effects of dataset characteristics on learning performance, it is also important to understand how individual intrinsic node attributes contribute to branching prediction. The intrinsic node representation defined in Eq. (1) combines spatial, structural, and geometric information, but the relative contribution of these attributes remains unclear. To investigate this, a controlled feature ablation study is conducted using progressively enriched subsets of the intrinsic representation.

1) Experimental setup: All configurations are evaluated using the synthetic tree dataset to isolate feature-related effects without reconstruction noise or structural inconsistencies present in real-world data. Uniform scaling is applied consistently across all configurations, and the original class distribution is preserved to maintain the natural branching statistics of the dataset.

Feature importance is evaluated using the branching classifier and the first regression model, which predicts the geometric attributes of the primary child branch. Although the complete framework includes a second regression model, it is omitted from this analysis since both regressors share the same architecture and learning formulation, and previous results showed consistent behavior between them.

Four feature configurations are considered, corresponding to progressively enriched intrinsic representations:

$$
\begin{aligned}
F_1&: \quad (p_i) \\
F_2&: \quad (p_i, o_i, d_i, d_i^{branch}) \\
F_3&: \quad (p_i, q_i, o_i, d_i, d_i^{branch}) \\
F_4&: \quad (p_i, q_i, r_i, r_i^{root}, o_i, d_i, d_i^{root}, d_i^{branch})
\end{aligned}
$$

representing spatial(F_1), spatial-structural(F_2), spatial-structural-orientation(F_3), and full intrinsic feature representations(F_4), respectively. These configurations enable systematic evaluation of how additional structural and geometric attributes influence both branching classification and geometric prediction.

2) Branching classification results: The branching classifier predicts the number of child nodes at each parent node, corresponding to zero, one, or two children. In this experiment, its performance is evaluated under different intrinsic feature configurations to analyze how feature composition influences the learning of branching topology.

Fig. 14 presents the confusion matrices obtained for the four feature configurations on the synthetic test dataset, while Table IV summarizes the corresponding quantitative performance in terms of mean accuracy and standard deviation across multiple runs with different random seeds.

When only spatial coordinates are used as input features, the classifier achieves limited predictive performance with a mean test accuracy of $55.55\% \pm 0.57$. The confusion matrix [Fig. 14(a)] shows that most nodes are predicted as continuation nodes (class 1), indicating that spatial position alone does not provide sufficient information to determine the branching topology. This result highlights that the branching state of a node is primarily governed by structural relationships within the tree rather than its absolute spatial location.

TABLE IV. CLASSIFICATION PERFORMANCE UNDER DIFFERENT INTRINSIC FEATURE SUBSETS (MEAN ± STANDARD DEVIATION)

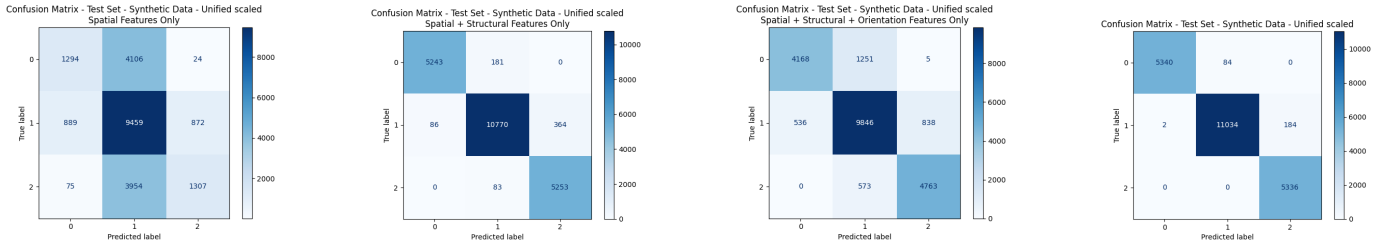
Feature Set	**Test Accuracy (%)**
Spatial Features	55.55 ± 0.57
Spatial + Structural Features	97.80 ± 0.80
Spatial + Structural + Orientation Features	96.12 ± 3.88
Full Intrinsic Feature Set	98.75 ± 0.06

When structural attributes are incorporated together with spatial features, classification performance improves significantly, achieving a mean test accuracy of $97.80\% \pm 0.80$. The corresponding confusion matrix [Fig. 14(b)] shows that all three branching categories are predicted with high reliability. Structural attributes such as branch order and relative distances provide important topological context that allows the network to distinguish between terminal nodes, continuation nodes, and bifurcation nodes. This demonstrates that structural descriptors capture the hierarchical organization of tree skeletons and play a critical role in learning branching patterns.

The addition of orientation information results in a slight reduction in mean classification accuracy to $96.12\% \pm 3.88$. As shown in Fig. 14(c), the classifier continues to capture meaningful structural patterns, although the increased variance suggests that orientation features introduce additional geometric variability that can affect training stability. While orientation encodes local branch direction, it does not directly determine the number of child branches, which explains its limited contribution to classification performance.

Finally, when the complete intrinsic feature representation is used, the classifier achieves the highest predictive performance with a mean test accuracy of $98.75\% \pm 0.06$. The confusion matrix [Fig. 14(d)] shows highly accurate classification performance across all branching categories. The inclusion of geometric attributes such as branch radius and cumulative distance from the root provides additional contextual information that stabilizes the learning process and allows the network to capture both structural and geometric relationships governing tree growth.

3) Regression results: The regression models are evaluated using the same four intrinsic feature configurations used in the classification experiment. The first regression network predicts the geometric attributes of the primary child branch, including branch orientation represented as a quaternion, as well as two geometric parameters: branch radius and edge length relative to the parent node. Fig. 15 presents the error distributions for both orientation and geometric predictions, while Table V



(a) Spatial features only

(b) Spatial + structural features

(c) Spatial + structural + orientation features

(d) Full intrinsic feature representation

Fig. 14. Confusion matrices for the branching classifier under different intrinsic feature subsets evaluated on the synthetic test dataset.

TABLE V. REGRESSION PERFORMANCE COMPARISON UNDER DIFFERENT INTRINSIC FEATURE SUBSETS USING REGRESSOR 1 (MEAN ± STANDARD DEVIATION)

Feature Set	Orientation Loss	Geometry Loss
Spatial	0.0215 ± 0.00001	$(11.2 \pm 0.45) \times 10^{-6}$
Spatial + Structural	0.0208 ± 0.00001	$(1.00 \pm 0.00) \times 10^{-6}$
Spatial + Structural + Orientation	0.0185 ± 0.0001	$(2.00 \pm 0.00) \times 10^{-6}$
Full Intrinsic Feature Set	0.0185 ± 0.0001	$(2.40 \pm 0.55) \times 10^{-6}$

summarizes the corresponding quantitative performance across multiple runs.

a) Orientation prediction: Fig. 15(a), (c), (e), and (g) shows the angular error distributions between the predicted and ground-truth branch orientations for the four feature configurations.

When only spatial coordinates are used as input [Fig. 15(a)], the orientation predictions exhibit a broad error distribution, with most predictions falling within the range of 10° to 30°. This indicates that spatial position alone provides limited information about local branch direction. Incorporating structural attributes [Fig. 15 (c)] results in a more concentrated distribution, suggesting that hierarchical information provides useful contextual cues for estimating branch orientation.

A more significant improvement is observed when orientation features are included [Fig. 15 (e)], where the angular errors become more tightly distributed. Using the full intrinsic feature representation [Fig. 15 (g)] produces the most stable predictions however, the improvement over the previous configuration is incremental. This behavior is consistent with the quantitative results in Table V, where orientation loss decreases steadily with feature enrichment and stabilizes when all features are included.

b) Geometric attribute prediction: Fig. 15(b), (d), (f), and (h) illustrates the prediction error distributions for branch radius and edge length. When only spatial coordinates are used [Fig. 15(b)], radius prediction exhibits a wide error distribution, indicating limited geometric information. In contrast, edge length predictions are relatively accurate even under this setting, suggesting partial correlation with spatial positioning. The addition of structural attributes [Fig. 15(d)] significantly improves both radius and length predictions, with errors becoming strongly concentrated within small percentage ranges.

When orientation information is incorporated [Fig. 15(f)], radius prediction improves slightly while edge length remains consistently accurate. The full intrinsic representation [Fig. 15(h)] produces the tightest radius error distribution, although edge length accuracy is marginally lower than the spatial-structural set. Because the aggregate geometry loss is dominated by edge length, Table V reports its lowest value for the spatial-structural configuration. Overall, the largest gain occurs when structural features are introduced; subsequent additions redistribute accuracy between geometric attributes rather than reducing error further.

4) Discussion: The results provide clear evidence of how different intrinsic feature groups contribute to learning tree branching structure. The feature ablation study shows that spatial coordinates alone are insufficient for reliably predicting either branching topology or branch geometry, as reflected in the lower classification accuracy and broader regression error distributions. This confirms that structural organization of tree skeletons cannot be inferred solely from global spatial position.

Structural attributes, including branch order and relative distances, emerge as the most influential features across both classification and regression tasks. Incorporating these descriptors leads to a substantial improvement in performance, indicating that hierarchical context plays a central role in determining both branching decisions and geometric properties. This highlights the importance of encoding topological relationships when modeling tree structures. Orientation features primarily enhance the regression task associated with predicting branch direction. Since orientation directly represents local geometric direction, it provides essential information for accurate quaternion prediction, resulting in more concentrated angular error distributions. However, its contribution to the classification task remains limited, as the number of child branches is governed more strongly by structural relationships than by local orientation.

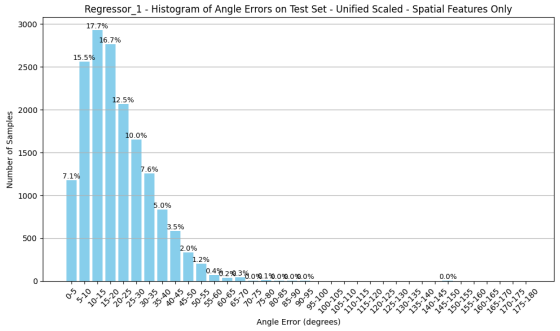
The full intrinsic feature representation yields the highest classification accuracy and the most stable orientation predictions, while geometric accuracy reflects a trade-off between radius and edge length rather than a uniform improvement. By combining spatial, structural, and geometric attributes, the model captures complementary aspects of tree structure, with different feature groups supporting different prediction tasks. These findings demonstrate that effective learning of tree branching behavior requires a balanced integration of hierarchical and geometric information, rather than reliance on any single feature group.

C. Perceptual Evaluation of Generated Tree Structures

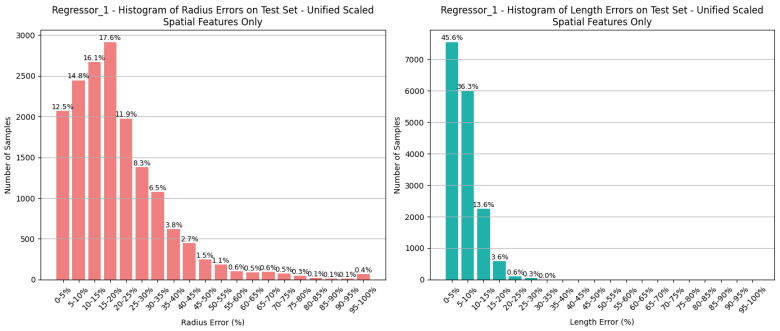
To assess whether the employed learning pipeline produces visually plausible tree structures, a user-based perceptual evaluation was conducted. In this context, realism does not refer to biological accuracy, but to whether a generated tree appears natural and believable within a virtual environment. The evaluation compares generated trees against dataset reference structures through pairwise preference trials.

1) Experimental setup: The study involved 25 participants, each completing 25 pairwise comparison trials, yielding 625 trials in total. In each trial, two tree structures were presented side by side and participants were asked to select the one that appeared more visually realistic. Trees were drawn from two pools of 20 instances each: a generated pool and a reference pool. Left-right placement was randomized to avoid positional bias.

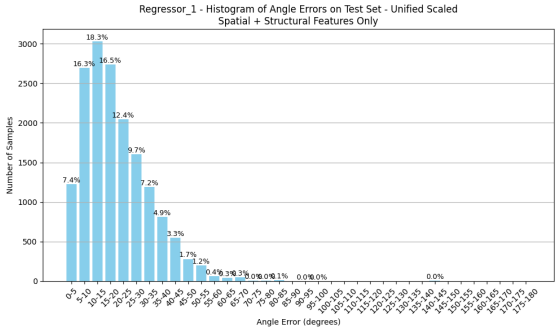
All generated trees were produced using the employed learning pipeline with uniform scaling, consistent with previous analyses. Since generation requires an initial root position, a fully automated root selection strategy was employed: multiple candidate positions were sampled within a predefined region and evaluated using structural criteria, including distance to the center of mass and spatial balance of node distribution. The candidate producing the most balanced structure was selected, ensuring reproducibility and eliminating manual intervention from the evaluation.



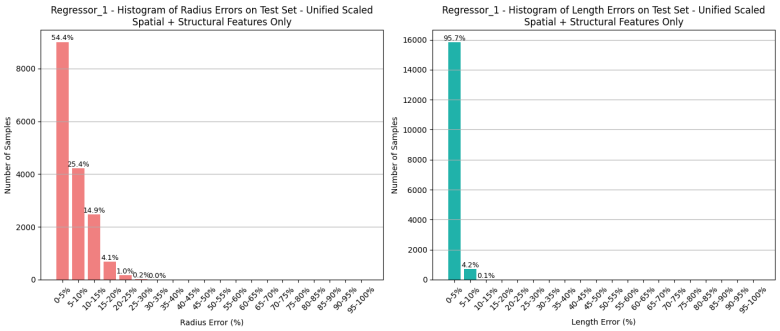
(a) Spatial features — Orientation



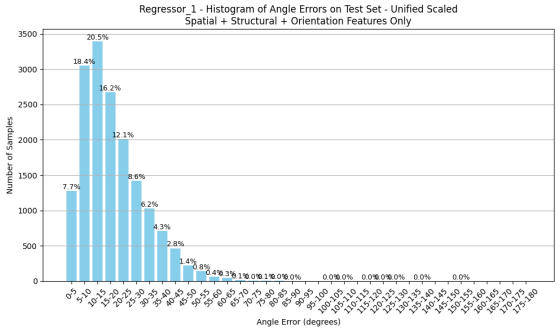
(b) Spatial features — Geometry



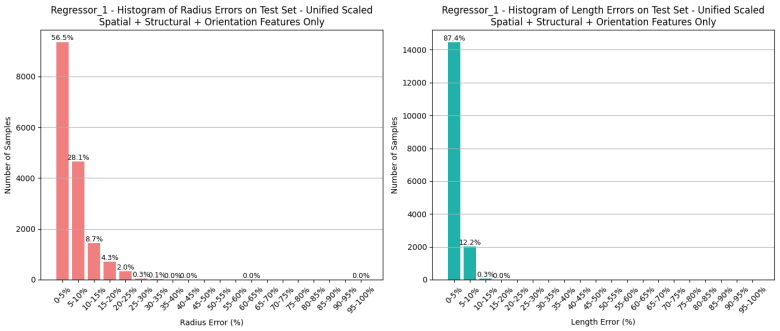
(c) Spatial + structural — Orientation



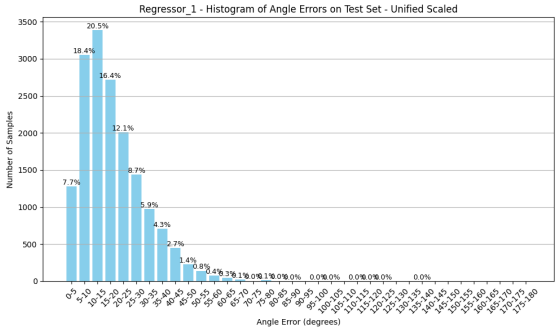
(d) Spatial + structural — Geometry



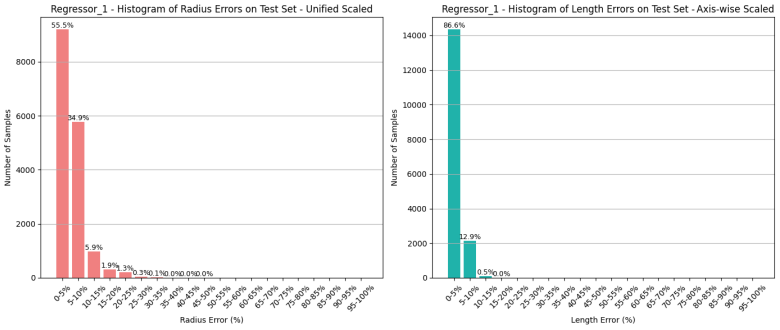
(e) Spatial + structural + orientation — Orientation



(f) Spatial + structural + orientation — Geometry



(g) Full intrinsic features — Orientation



(h) Full intrinsic features — Geometry

Fig. 15. Error distributions for the regression model under different intrinsic feature subsets. In each row, the left image shows angular error distributions for orientation prediction, while the right image shows radius and edge length error distributions. From top to bottom, the rows correspond to progressively enriched feature representations: spatial features only, spatial with structural features, spatial with structural and orientation features, and the full intrinsic feature representation. The results illustrate that incorporating additional intrinsic attributes leads to more concentrated error distributions and improved prediction accuracy.

Fig. 16. Overall selection rates across all 625 trials.

2) Results and analysis: The overall selection rates are presented in Fig. 16. Across all 625 trials, participants selected reference trees in 66.6% of cases and generated trees in 33.4% of cases. A binomial test against a 50% random baseline confirms this difference is statistically significant, indicating that participants could consistently distinguish between the two types of structures.

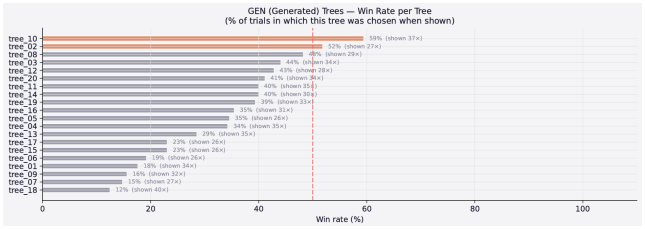
While reference trees were generally preferred, generated trees were selected in approximately one-third of trials, which represents a non-trivial proportion. Per-tree win rates, shown in Fig. 17, reveal considerable variation across individual generated instances, with some trees achieving selection rates of 40–59%, approaching the perceptual quality of reference structures, while others fall in the 12–20% range. A similar variation is observed among reference trees, suggesting that even dataset structures differ in perceived visual balance.

3) Discussion: The results indicate that the learning pipeline captures key intrinsic branching characteristics, producing perceptually plausible structures in a meaningful proportion of trials. However, the consistent preference for reference trees and variability across generated instances highlight a remaining quality gap. This variability is partly attributable to the automated root selection process, which ensures fairness and reproducibility but does not guarantee globally optimal structural balance. Overall, the findings confirm the model's ability to generate visually plausible structures while motivating further work on improving generation consistency and initialization strategies.

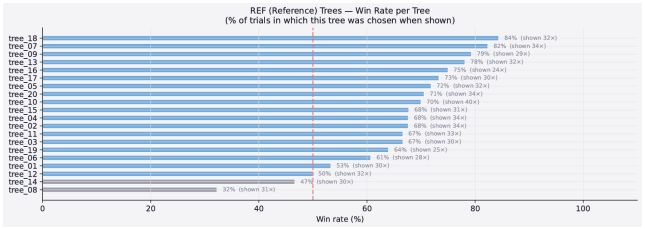
VII. Discussion: Towards a Hybrid Pipeline

The experiments presented in this work provide a consistent picture of the capabilities and limitations of learning intrinsic tree branching behavior from data. Under controlled conditions, the employed learning pipeline accurately learns local branching topology and geometric relationships, as demonstrated by the high performance achieved on synthetic tree structures. The feature ablation study further shows that structural and geometric attributes play a critical role in enabling this learning process, confirming that intrinsic node representations can effectively encode local branching behavior. The perceptual evaluation additionally demonstrates that generated structures are perceived as visually plausible in a meaningful proportion of trials, providing human-centred evidence that the model captures key intrinsic branching characteristics beyond quantitative metrics alone.

However, the results obtained using real-world tree data reveal important limitations. While the models remain stable



(a) Generated trees



(b) Reference trees

Fig. 17. Per-tree win rates for generated and reference tree structures.

and continue to capture meaningful structural patterns, prediction accuracy decreases due to the increased variability and noise present in reconstructed tree skeletons. This degradation is also reflected at the perceptual level, where generated trees were consistently preferred less than reference structures and notable variation in quality across individual generated instances was observed. These findings suggest that purely data-driven learning of intrinsic branching behavior is sensitive to the quality and consistency of the underlying geometric representation.

Taken together, these observations indicate that learning-based approaches are well-suited for modeling intrinsic branching relationships when reliable structural cues are available, but may struggle to fully account for irregularities and ambiguities present in real-world data. This naturally motivates a hybrid modeling strategy that separates intrinsic structure learning from external influences and reconstruction noise.

In such a hybrid pipeline, the learning-based component can be used to model local branching decisions and geometric relationships based on intrinsic node attributes, while complementary algorithmic or rule-based methods can be employed to enforce global structural constraints, incorporate environmental effects, or regularize noisy inputs. This decoupled approach leverages the strengths of data-driven learning for capturing complex local patterns, while maintaining robustness and control through explicit modeling of factors that are difficult to learn directly from data.

Overall, the results of this study suggest that combining learning-based intrinsic modeling with structured procedural or physics-inspired methods provides a promising direction for generating visually plausible and structurally consistent tree models in practical applications.

VIII. Limitations and Future Work

While the employed learning pipeline successfully learns intrinsic tree branching behavior from data, several limitations remain that highlight opportunities for further improvement.

First, the secondary branch prediction task, handled by Regressor 2, consistently exhibits higher errors than the primary branch model, reflecting greater structural variability of secondary branches. Also because evaluation is node-level, how such local errors accumulate during recursive generation was not quantified, and the relationship between local accuracy and global coherence at increasing depth remains uncharacterized.

Second, while a fully automated root selection strategy is employed to determine suitable initialization positions, the approach relies on structural heuristics that do not guarantee globally optimal balance across all generated instances. Some trees exhibit spatial asymmetries that affect perceptual quality, indicating that more principled initialization methods warrant further investigation.

Third, the availability of suitable real-world datasets poses a practical limitation. The employed approach relies on tree representations in the form of explicit skeleton graphs with associated geometric attributes. In this work, the TreeML-Data dataset [13] is used as the primary source of real-world tree structures, as it provides reconstructed tree models with hierarchical connectivity derived from terrestrial LiDAR scans. However, such datasets remain limited in availability, and existing real-world tree models are typically reconstructed through scanning and skeletonization processes that introduce noise, missing structures, and geometric inconsistencies. This restricts the diversity and quality of training data available for learning intrinsic branching patterns.

These limitations suggest several directions for future work. One important direction is to improve the modeling of secondary branch geometry, for example by exploring alternative formulations that better capture the dependencies and variability associated with bifurcation structures. Another promising direction is the development of more principled root initialization methods, enabling the generation process to achieve more consistent structural quality. Improving robustness to real-world data remains an additional challenge, which could be addressed through enhanced preprocessing techniques, noise-aware learning strategies, or the incorporation of more diverse and higher-quality datasets. Finally, integrating the learned intrinsic growth model with explicit environmental simulations represents a promising direction for producing more realistic and adaptive tree structures.

IX. Conclusion

This work presented a behavioral analysis of an employed learning pipeline for intrinsic 3D tree branching, based on local node attributes and iterative structure generation. Rather than targeting biologically accurate growth, the focus was on learning structurally consistent and visually plausible branching patterns suitable for virtual environments.

The experimental results demonstrate that intrinsic branching relationships can be learned effectively under controlled conditions. Models trained on synthetic tree structures achieve high classification accuracy and low regression errors, indicating that the intrinsic feature representation successfully captures the local geometric and structural dependencies governing branching behavior. The feature ablation study confirms that structural attributes play a critical role in learning branching topology, while the perceptual evaluation demonstrates that these quantitative improvements translate to human-perceived realism in a meaningful proportion of trials.

When applied to real-world tree data, the learning problem becomes more challenging due to noise, reconstruction artifacts, and structural variability. Although meaningful branching patterns are still learned, prediction accuracy decreases and perceptual quality remains below that of reference structures, highlighting the sensitivity of the approach to data quality and representation consistency.

Overall, the findings suggest that intrinsic tree growth can be effectively modeled as a local learning problem, provided that reliable structural information is available. These results support the use of data-driven methods for modeling intrinsic branching behavior and motivate the development of hybrid pipelines, where learned intrinsic growth models are combined with explicit environmental simulations to produce more realistic and adaptable tree structures.

Acknowledgment

We want to thank the University of Colombo School of Computing for providing the academic environment that supported this research.

Declaration on Generative AI

Generative AI tools were used to assist in language refinement and improvement of clarity during the preparation of this manuscript. All technical content, experimental design, analysis, and conclusions were developed and verified by the authors. The authors take full responsibility for the accuracy and integrity of the work.

References

[1] X. Zhou, B. Li, B. Benes, S. Fei, and S. Pirk, "Deeptree: Modeling trees with situated latents," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, pp. 5795–5809, 2023.

[2] Z. Lam and S. A. King, "Simulating tree growth based on internal and environmental factors," in *Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, GRAPHITE '05, (New York, NY, USA), p. 99–107, Association for Computing Machinery, 2005.

[3] B. Benes, "Visual model of plant development with respect to influence of light," in *Computer Animation and Simulation*, 1997.

[4] A. Runions, B. Lane, and P. Prusinkiewicz, "Modeling trees with a space colonization algorithm," in *Eurographics Workshop on Natural Phenomena*, 2007.

[5] W. Pałubicki, K. Horel, S. Longay, A. Runions, B. Lane, R. Měch, and P. Prusinkiewicz, "Self-organizing tree models for image synthesis," *ACM SIGGRAPH 2009 papers*, 2009.

[6] S. Pirk, O. Stava, J. Kratt, M. A. M. Said, B. Neubert, R. Měch, B. Benes, and O. Deussen, "Plastic trees: interactive self-adapting botanical tree models," *ACM Trans. Graph.*, vol. 31, July 2012.

[7] B. Beneš, N. Andrysco, and O. Šťava, "Interactive modeling of virtual ecosystems," in *Proceedings of the Fifth Eurographics Conference on Natural Phenomena*, NPH'09, (Goslar, DEU), p. 9–16, Eurographics Association, 2009.

[8] B. Tóth and S. Szénási, "Tree growth simulation based on ray-traced lights modelling," *Acta Polytechnica Hungarica*, 2020.

[9] O. Stava, S. Pirk, J. Kratt, B. Chen, R. Měch, O. Deussen, and B. Benes, "Inverse procedural modeling of trees," 2014.

[10] H. Wang, B. Zhang, J. Klein, D. L. Michels, D. Yan, and P. Wonka, "Autoregressive generation of static and growing trees," *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, 2025.

[11] J. J. Lee, B. Li, and B. Benes, "Latent l-systems: Transformer-based tree generator," *ACM Transactions on Graphics*, vol. 43, pp. 1 – 16, 2023.

[12] X. Zhou, B. Li, B. Benes, A. Habib, S. Fei, J. Shao, and S. Pirk, "Treestructor: Forest reconstruction with neural ranking," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 63, pp. 1–19, 2025.

[13] H. Yazdi, Q. Shu, T. Rötzer, F. Petzold, and F. Ludwig, "A multilayered urban tree dataset of point clouds, quantitative structure and graph models," *Scientific Data*, vol. 11, 2024.

[14] Z. Liu, Y. Li, F. Tu, R. Zhang, Z. Cheng, and N. Yokoya, "Deeptreesketch: Neural graph prediction for faithful 3d tree modeling from sketches," *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 2024.