Pacific Graphics 2024
R. Chen, T. Ritschel, and E. Whiting
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 7

VRTree: Example-Based 3D Interactive Tree Modeling in Virtual Reality

Di Wu[1,2], Mingxin Yang[1,2], Zhihao Liu[3], Fangyuan Tu[1,4], Zhanglin Cheng[1†]

[1] Shenzhen Key Laboratory of Visual Computing and Analytics, SIAT, Chinese Academy of Sciences, China.
[2] University of Chinese Academy of Sciences, China. [3] The University of Tokyo, Japan.
[4] Japan Advanced Institute of Science and Technology, Japan

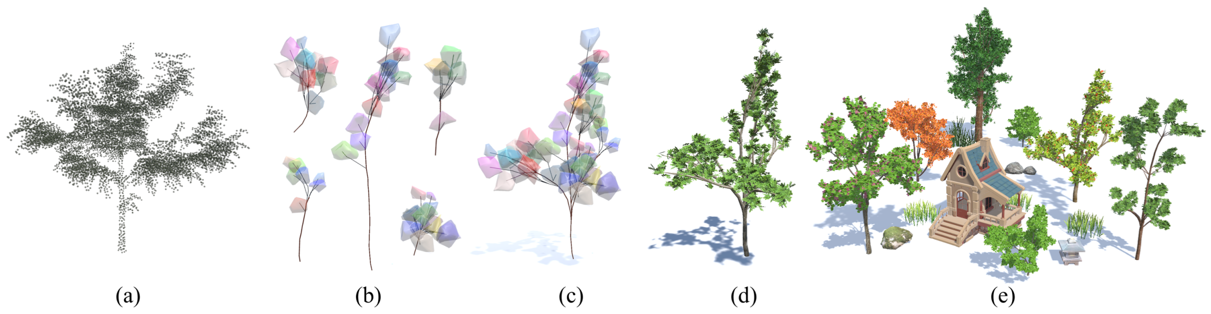


Figure 1: *Our method automatically decomposes real tree point clouds (a) into a set of structural examples represented as hierarchical branches and lobes (b). Users can then interactively assemble these examples according to their intentions (c), allowing the efficient generation of a 3D tree model with high realism (d). An outdoor scene comprising tree models created by our system is shown (e).*

Abstract

We present VRTree, an example-based interactive virtual reality (VR) system designed to efficiently create diverse 3D tree models while faithfully preserving botanical characteristics of real-world references. Our method employs a novel representation called Hierarchical Branch Lobe (HBL), which captures the hierarchical features of trees and serves as a versatile intermediary for intuitive VR interaction. The HBL representation decomposes a 3D tree into a series of concise examples, each consisting of a small set of main branches, secondary branches, and lobe-bounded twigs. The core of our system involves two key components: (1) We design an automatic algorithm to extract an initial library of HBL examples from real tree point clouds. These HBL examples can be optionally refined according to user intentions through an interactive editing process. (2) Users can interact with the extracted HBL examples to assemble new tree structures, ensuring the local features align with the target tree species. A shape-guided procedural growth algorithm then transforms these assembled HBL structures into highly realistic, fine-grained 3D tree models. Extensive experiments and user studies demonstrate that VRTree outperforms current state-of-the-art approaches, offering a highly effective and easy-to-use VR tool for tree modeling.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ ***Shape modeling; Virtual reality;*** *Graphics systems and interfaces;*

1. Introduction

Among common objects, botanical trees are notably challenging to create due to their geometrical complexity. The primary difficulty arises from the intricate branch structures and complex foliage details, resulting in the absence of easy-to-use systems for modeling trees, particularly for novice users. With the rapid growth of virtual reality (VR) technologies, a series of immersive interaction methods [LSL*19, YH21, ZLC*21, LZC21] have been introduced for 3D content creation, which may open up new possibilities for in-

† Corresponding author: zl.cheng@siat.ac.cn

teractive tree modeling. However, directly adapting traditional 2D interactive approaches (e.g., sketching [IOI06a, IOI06b, WBCG09, NGDA*16, KYC*17, LZZ*21, JZF*21]) in VR environments, will pose numerous new challenges. For example, the low positioning precision of VR devices hinders the accurate creation of complex objects like trees. Furthermore, novice users often lack the botanical knowledge necessary to create tree models that faithfully reflect real-world characteristics. On the other hand, example-based 3D creation techniques have recently emerged to streamline the interactive 3D modeling process [CKGK11, KCKK12, DXA*19, CRW*20, YCC*20, FXX*22]. These methods employ a semantic decomposition strategy to synthesize new 3D objects by assembling distinct example parts from a pre-generated library. By adopting this paradigm, the complex structural priors can be implicitly encapsulated within these predefined parts, allowing users to circumvent the burden of learning specialized knowledge, and effectively simplifying the originally tedious modeling process into a more manageable task of part assembly. Therefore, we extend the methodology of example-based generation to propose a novel VR interactive system that enables novice users to intuitively and effortlessly design realistic 3D tree models with high fidelity and realism.

To begin, it is essential to identify an appropriate intermediate representation that not only effectively depicts the botanical structure of 3D trees but also facilitates example-based interactive modeling. Traditionally, point clouds often serve as a predominant 3D representation in tree modeling due to their ease of acquisition. However, point clouds are typically unsuitable for interactive systems since they are unstructured and prone to noise. Alternatively, some researchers simplify 3D trees into skeletal graphs [XGC07, LYO*10], which reduces the structural complexity of trees compared to point clouds. However, this representation is still too complicated for interactive use since fully creating millions of branches for dense foliage is very time-consuming. More recently, Livny *et al.* [LPC*11] introduce a lightweight representation termed texture-lobes for static point cloud reconstruction, which abstracts the foliage details into a series of coarse-grained canonical geometry structures. We argue that this new representation effectively encodes tree structures while preserving significant botanical features, making it a simple yet intuitive medium for interactive modeling. Inspired by this, Zhang *et al.* [ZLC*21] make an initial attempt to construct 3D tree models by manually drawing texture-lobes in mid-air. However, using imprecise VR devices to craft these lobes with free-form shapes fully from scratch remains labor-intensive and results in a suboptimal user experience. In addition, manual drawing without real reference diminishes one of the key advantages of this representation: accurately capturing the botanical characteristics of trees.

In this work, we initially extend the texture-lobes representation to a hierarchical multi-level representation encompassing main branches, secondary branches, and lobe-bounded twigs, dubbed Hierarchical Branch Lobe (HBL), enabling a seamless integration into the example-based modeling paradigm. Subsequently, we also develop an automatic module that extracts the HBL structure and decomposes it into individual HBL examples from real-world tree point clouds. Moreover, a refinement module is selectively introduced to allow users to fine-tune and edit individual HBL examples with minimum interaction. Our proposed framework facilitates the creation of a comprehensive Multi-level Tree Examples Library (MTEL), comprising versatile HBL examples derived from a wide range of 3D tree point clouds across various species. During the modeling process, the library guides the modelers with underlying botanical knowledge by suggesting HBL examples that contain realistic branch structures and foliage distribution patterns. This approach relieves users from the need to acquire detailed botanical knowledge or draw precise 3D strokes in mid-air, allowing them to construct highly realistic 3D tree models with minimal effort. Finally, fine-grained details of the generated trees, such as twigs and foliage, are synthesized using a shape-guided procedural growth algorithm. A typical example of our holistic pipeline is illustrated in Fig. 1. Starting with a single tree point cloud (a), a sub-library including diverse HBL examples is constructed (b). Leveraging the HBL sub-library, trees with novel HBL structures can be generated by simply assembling the HBL examples in a VR environment (c). Followed by an automatic multi-level branch merging process and a procedural growth algorithm, a visually appealing 3D tree model with faithful botanical features can be synthesized (d), which can be further incorporated into realistic outdoor scenes (e).

In summary, our contributions include:

- An example-based tree modeling approach that can create diverse and realistic tree models from a limited number of real tree examples, even a single example, while preserving the intricate structural details of the real examples.
- A Hierarchical Branch-Lobe (HBL) representation that serves as an effective and efficient intermediary for tree creation in VR, allowing the assembly of new tree models by combining HBL examples and significantly reducing the need for tedious user interactions.
- A set of novel 3D interactive tools specifically designed for tree modeling within VR environments, including branch manipulation, lobe merging, and lobe splitting, significantly enhancing the efficiency and ease of use in 3D tree modeling.

2. Related Work

Automatic tree modeling. 3D tree modeling is a longstanding topic in the computer graphics community. To facilitate realistic tree models, data-driven modeling techniques are proposed to recover the branch structure and foliage distribution from the multiple input sources (e.g., 3D point clouds [XGC07, LYO*10, LPC*11, DLL*19, LGB*21], images [GXY*18, LKK*21, LWG*21, LLB*24], etc.). However, it becomes significantly challenging due to the inherently ill-posed nature of the problem when dealing with vague semantics in the input sources. Therefore, conventional automatic tree modeling approaches are proposed by drawing upon either the botanical prior, statistical prior or a combination thereof (L-system [Lin68, Pru86, PL12], Space-colonization [RLP07, PHL*09, YLG*18] and Particle-flow [NFD07]). For more comprehensive information about 3D modeling and reconstruction of plants and trees, please refer to the surveys by Okura *et al.* [Oku22] and Cardenas *et al.* [CDOFJ22].

Conventionally, branch-level skeletal graph (BSG) is the most common intermediate representation due to its intuitive utility.

Livny *et al.* [LYO*10] introduce a heuristic directional-field optimization algorithm to extract the BSG from the tree point clouds. AdTree [DLL*19] extends the idea by incorporating a specific simplification strategy to maintain branch structure. Benefiting from the development of deep point cloud networks, TreePartNet [LGB*21] applies a deep neural network that decomposes tree point clouds towards local cylindrical shapes and extracts the local medial axis as the tree skeleton. Wang *et al.* [WLX*18, WLJ*18] encode the tree branches into a shared latent shape space with the corresponding metric, upon which they develop a tree generative approach. Dobbs *et al.* [DBGA23] also introduce a neural extraction network and propose a multi-species synthetic tree dataset. Li *et al.* [YZB*24] leverage a voxel diffusion model [HJA20] to reconstruct the 3D BSG model from single images. However, BSG lacks the capability to grasp the foliage distribution. For this purpose, several novel representations are proposed. Livny *et al.* [LPC*11] present a texture-lobes model to capture both the branch structure and the foliage distribution and further develop a proxy extraction approach based on [LYO*10]. Li *et al.* [LKK*21] utilize a cylinder-based proxy and apply a botanical growing algorithm upon the proxy to further generate a full tree. Neural latent representation is introduced by Lee *et al.* [LLB23] and Zhou *et al.* [ZLB*23] to synthesize novel 3D trees. In another representative line of research, Argudo *et al.* [AAC20] propose an image-based tree billboards modeling approach, synthesizing a large amount of tree billboards from a few exemplars. Xie *et al.* [XYS*15] adopt an example database comprising real-life trees, treating tree modeling as a mesh completion task. Though analogous to ours, their synthesized tree geometry is constrained by the representation capacity of the manually-collected dataset, whereas ours achieves automatic extraction and interactive post-editing.

Interactive tree modeling. Despite the progressive endeavors of automatic tree modeling, the semantic ambiguity induced by noisy point clouds still poses a great challenge in modeling fidelity, which often leads to noticeable artifacts in branch structure. Therefore, interactive tree modeling appears to be a more elegant proposal, introducing human effort to simplify the semantic ambiguity of 3D point clouds. Among various interaction tools, freehand sketch is one of the most frequently used methods to depict tree branch structure. One example is Longay *et al.* [LRBP12], which combines sketch drawing with procedural modeling to produce a complete 3D tree. Following the idea, Manfredi *et al.* [MCEG23] present a deep neural network that maps the stylized sketches towards a tree parametric model proposed by Weber *et al.* [WP95]. To enable more realistic-look modeling from under-constrained freehand sketches, researchers also aim to produce more faithful 3D branching structures based on probabilistic optimization [CNX*08] or deep learning [LLT*24] techniques. However, the branch structure, especially secondary branches, is often occluded by the densely distributed foliage, hindering unprofessional user sketching properly. Furthermore, depth ambiguity existed in 2D sketches constraints its ability to synthesize fully specified trees.

With the rapid advancement in consumer VR devices, 3D immersive interaction has been incorporated into tree modeling. To explore the use of 6-DOF motion, Liu *et al.* [LSL*19] present an interactive system for modeling trees using 3D gestures in a VR platform. By lifting 2D sketch to 3D, Yuan *et al.* [YH21] successively develop sketch-based modeling in immersive VR environments. They also integrate the system with algorithms for generating branch and twig geometry. A more recent work related to ours is Zhang *et al.* [ZLC*21], which takes a mid-air finger sketching scheme in a VR scenario and takes advantage of texture-lobes representation [LPC*11]. However, the requirement to draw 3D sketches for both branches and foliage lobes deprives their easy-to-use characteristics in intuitive user interaction. Besides, the constrained representation capacity of user-sketched lobes further renders their synthesized 3D trees notably unrealistic. Taking advantage of the proposed Multi-level Tree Examples Library (MTEL), we are able to effectively generate high-fidelity 3D trees with trivial interaction.

3. Method

3.1. Overview

The design goals of VRTree lie in two-fold:

- To construct an HBL (Hierarchical Branch Lobe) examples library from existing tree point clouds, facilitating further tree generation.
- To synthesize novel 3D tree models not only reflecting the user's imagination but also preserving faithful botanical features.

The pipeline of our interactive system is illustrated in Fig. 2. We begin by introducing the HBL (Hierarchical Branch Lobe) structure in Section 3.2, laying the foundation of our work. Leveraging the HBL representation as an intermediary, we then present the devised interactive approach at both the point cloud and HBL levels in Section 3.3. The two key stages—automatic decomposition, interactive refinement from point clouds to HBL structure, and interactive example-based tree generation—are detailed in Section 3.4 and Section 3.5 respectively.

3.2. Hierarchical Branch Lobe (HBL)

To facilitate user-friendly interaction in VR, we introduce a novel intermediate representation called the Hierarchical Branch Lobe (HBL). This structure comprises main branches, secondary branches, and lobe-bounded twigs. Specifically, the main branches define topological structures and primary shape of the tree's main trunk, while secondary branches represent the branch structure extending from the main trunk to the tree crown. Finally, the tree crown's appearance is determined by the spatial distribution of lobes, from which twigs and leaves proliferate. As shown in Fig. 2 (d), a set of main branches (red curves), secondary branches (black curves), and lobes (various colored geometry) is organized into a hierarchical multi-level structure.

In contrast to traditional 3D tree intermediate representations [Lin68, Pru86, SJJ09, QQJ11, PL12], which often rely on iterative growing paradigms, our HBL supports a tree synthesizing framework that assembles multiple HBL examples selected from the proposed library by users. This approach allows for independent control over distinct levels (main branch, secondary branch, lobe-bounded twigs level), enabling flexible user interactions.

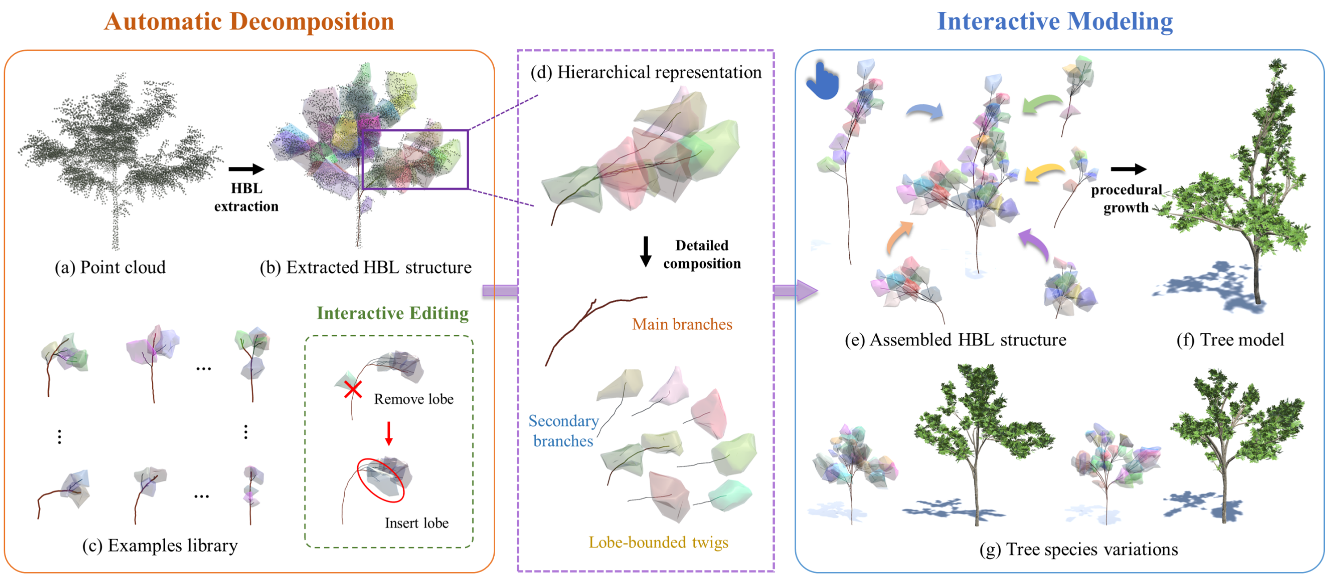


Figure 2: *Overview of our interactive tree modeling system. Automatic decomposition (a-d): Given the point cloud of a specified tree (a), we first extract the Hierarchical Branch Lobe (HBL) structure (b), then decompose the HBL structure into a series of examples to construct a library (c). The interactive editing module enables users to further refine HBL examples. Each HBL example comprises a set of main branches, secondary branches, and lobe-bounded twigs (d). Interactive modeling process (e-g): Through assembling examples in the library, we recompose a new HBL structure (e), and then a procedural growth algorithm is applied to generate a final model with twigs and leaves (f). Guided by the HBL examples, tree species variations can be generated (g).*

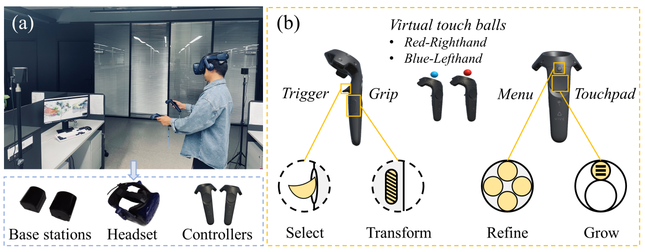


Figure 3: *User interaction design. (a) User interface and VR devices. (b) Controller buttons.*

3.3. User Interaction Design

As shown in Fig. 3, our VR devices consist of a headset, two base stations, and two controllers. To facilitate interaction, we add two visual touch balls, colored blue and red, on the controllers, representing the left and right hand respectively. When the user touches with an HBL example using the visual controller, the branches and lobes are highlighted with the same color as the touch balls, making it easy to distinguish different levels of interaction. Moreover, four commonly used buttons are employed to execute user commands: *grip, trigger, touchpad, menu*. The *grip* button is used to translate and rotate entire HBL examples or individual lobes, the *trigger* button involves a short press for selection and a long press for sketching (to create new HBL examples), and the *touchpad* and the *menu* buttons handle boolean operations such as merging or splitting lobes.

All interactions are categorized into two types: refinement and modeling, corresponding to the HBL decomposition from point clouds and the example-based tree modeling, respectively. In cases where the automatic algorithm (introduced in Section 3.4) fails to appropriately decompose HBL examples for extremely complicated instances, the VR interface allows users to easily identify and refine the decomposition result from a 360-degree view. On the modeling side, users can translate, rotate, and sketch to assemble existing HBL examples from the library or draw new examples with just a few strokes, tailoring them to their creative intentions. To ensure the reliability of the proposed HBL examples library (MTEL), we meticulously examine each automatically decomposed example and enhance those of inferior quality.

3.4. Decomposition from Point Clouds

Automatically extracting an intermediate tree representation from point clouds has been a well-researched topic in recent years [XGC07, LYO*10, LPC*11, GXY*18, DLL*19]. However, these efforts primarily focus on extracting simplistic branch structures or basic texture lobes from point clouds, often lacking multi-level decomposition capabilities. Consequently, existing techniques are not directly suitable for our HBL extraction process. To address this, we introduce a two-stage HBL decomposition pipeline that combines an initial automatic decomposition module with optionally interactive assistance, resulting in satisfactory results for

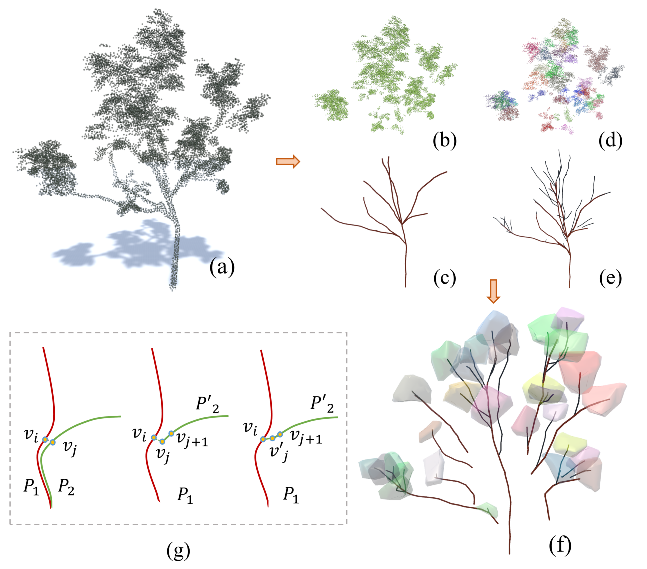


Figure 4: *Automatic decomposition of HBL examples from the 3D tree point cloud. (a) Point cloud. (b) Leaf points. (c) Main branches. (d) Clustering points for lobe-bounded twigs. (e) Main branches and secondary branches. (f) Full HBL structure. (g) Branch merging and smoothing.*

most instances. Our method is also effective in handling trees with incomplete point clouds, which are common in laser scanning or image-based reconstruction.

Automatic decomposition of HBL structure. We decompose the 3D tree point clouds into a series of HBL examples according to the semantic hierarchy pre-defined in our HBL structure, namely main branches, secondary branches and lobe-bounded twigs. As shown in Fig. 4, we demonstrate the specific procedure involved in our method. (a-b) Inspired by Guo *et al.* [GXY*18], we first leverage a heuristic segmentation scheme to separate foliage points from branch points. (a-c) To extract branches from the point cloud, we build a graph based on the K-nearest neighbor algorithm, weighting all the edges using the Euclidean distance, and computing all shortest paths from the tree root to other points. After that, we conduct a Farthest Point Sampling (FPS) on the raw point cloud (a) to obtain a set of points preserving the overall appearance of a tree. From the downsampled set of points, we attain the shortest paths terminating at these points and conduct a further merging process (see Algorithm 1) to extract the **main branches**. (b-d) Subsequently, K-means clustering is performed on the foliage points to acquire the clustering point for lobe-bounded twigs. (c-e) Using these generated lobes, we compute the shortest paths ending with the center of the lobes. After additional processing (see Algorithm 1), the **secondary branches** are constructed. (d-f) In parallel, by applying the α-Shape algorithm on the point clusters, we generate the canonical geometry for foliage lobes, upon which **lobe-bounded twigs** grow. (e-f) Finally, the full HBL structure of the tree is obtained.

The branch processing algorithm mentioned in (c) and (e) involves two steps: first, it identifies joint points where the main branches connect, and second, it determines joint points where the secondary branches connect to the main branches. The algorithm merges and processes branches based on their similarity, consistently combining the two most similar branches to maintain the compactness of the HBL example structure. The specific details are illustrated in Algorithm 1, where M and S denote the sets of main branches and secondary branches respectively, each of which is initially represented as shortest paths. P_i denotes the i-th shortest path.

Algorithm 1 Branch Merge Algorithm

```
 1: Step 1. Merge main branches
 2: Input:
 3:     Initialize a set $M$, let $M = P_l$, $P_l$ is the longest main branch.
 4:     Set $N_m = |M|$ as the total number of main branches;
 5: for $i$ in range(1, $N_m - 1$) do
 6:     Read the current branch $P_i$;
 7:     Load all branches in $M$;
 8:     Compute the similarity between $P_i$ and $P$ in $M$;
 9:     Search the most similar branch denoted as $P_{opt}$;
10:     Merge similar edges between $P_i$ in $P_{opt}$;
11:     Store the first non-merged point of $P_i$ as connection point;
12:     Update $M$ by adding processed $P_i$.
13: end for
14: Step 2. Merge secondary branches
15: Input:
16:     Initialize a set $S$, let $S$ contain all secondary branches
17:     Set $N_s = |S|$ as the total number of secondary branches;
18: for $j$ in range(1, $N_s$) do
19:     Read the current branch $P_j$ in $S$;
20:     Load all main branches in $M$ from Step 1;
21:     Compute the similarity between $P_j$ and $P$ in $M$;
22:     Search the most similar branch denoted as $P_{opt}$;
23:     Merge similar edges between $P_j$ in $P_{opt}$;
24:     Store the first non-merged point of $P_j$ as connection point.
25: end for
```

Note that in the above algorithm, a key aspect is accurately measuring branch similarity. Inspired by Shape Contexts [BMP02], a state-of-the-art method for computing shape similarity in object recognition and matching, we treat branches as 3D shapes and address the similarity measurement of branches as an optimal assignment problem. The cost $\sum_i \sum_j C[i,j]X[i,j]$ of the best assignment between vertices of two branches is used to measure their similarity, where $C[i,j]$ is the cost of matching the vertex of the first branch and the vertex of the second branch, and X is a boolean matrix. A lower cost indicates a higher degree of similarity between the two branches. Once the two most similar branches are identified, we traverse each pair of corresponding points with the same index. If the Euclidean distance between them is below a user-specified threshold, the corresponding edges are merged. If the distance exceeds this threshold, the point is treated as a connection point, as shown in Fig. 4 (g), where P_1 is the optimal main branch for secondary branch P_2 to merge with. We compute v_i as the connection point, and the points before v_j on the secondary branch are merged. A new edge between v_i and v_j is then built. Finally, drawing on insights from AdTree [DLL*19], we optimize the position of the connection point v_j by calculating the weighted average of the lengths from v_j to v_i and v_{j+1}:

$$v_j^{'} = \frac{v_i l_i + v_{j+1} l_{j+1}}{l_i + l_{j+1}} \tag{1}$$

where $v_j^{'}$ is the renewed point position, l_i and l_{j+1} are the lengths that v_i and v_{j+1} connected to the old position v_j. This update rule ensures the smoothness of the local branch within the HBL structure. By conducting this scheme on branches, we achieve a complete decomposition of HBL examples from the raw point cloud.

Interactive refinement of HBL example. To improve the robustness of our system in addressing intricate scenarios, we design and implement an interactive refinement module to address cases where automatic decomposition might be inadequate. Previous methods focus on inferring parameters and patterns from sketches [MCEG23] or pixel images [GJB*20] to generate new tree models. However, utilizing and adjusting these abstract tree parameters can be challenging, especially for novice users. Instead, visually intuitive and user-friendly tree examples are easier to manage. Moreover, users benefit from a 360-degree view in VR environments, allowing them to effortlessly remove or insert specific branches. Therefore, this interactive refinement of HBL examples can minimize the need for laborious manual parameter adjustment, significantly enhancing the overall user experience in the tree modeling process. In detail, we devise four operations specially tailored for refining HBL examples in a VR environment: *remove lobe, insert lobe, merge lobe, split lobe*. These operations not only facilitate the refinement process but also contribute to the construction of the examples library, greatly reducing the effort required to build the MTEL, which includes hundreds of HBL examples across various tree species.

- **Remove lobe**: delete the lobes with irregular shapes
- **Insert lobe**: select points to generate new lobes
- **Merge lobe**: merge lobes with small size but close location
- **Split lobe**: split lobes with big size against others

The implementation of these operations involves re-clustering lobe points and regenerating secondary branches. For instance, when lobes are merged into a single one, the backend algorithm regenerates a new canonical geometry using the α-Shape method. The secondary branch is then regenerated and optimized using Algorithm 1, connecting the pre-built main branch structure to the newly synthesized lobe. Fig. 5 demonstrates the consecutive refinements applied to an HBL example through these four operations. For example, (a) to (b) shows the merging of selected lobes, while (c) illustrates how a user can split a lobe by drawing strokes, which divide the corresponding point cloud into several parts. The number of strokes determines the number of new lobes, with sampled points from these strokes serving as initial clustering centers for the K-means algorithm. After separation, the α-Shape algorithm generates new lobe shapes from the separated parts of the point cloud, and the shortest paths from the tree root to each lobe's center define new branches, followed by a merging process using Algorithm 1. The operations for removing and inserting a lobe are demonstrated in (d) and (e) respectively. Additional auxiliary operations such as

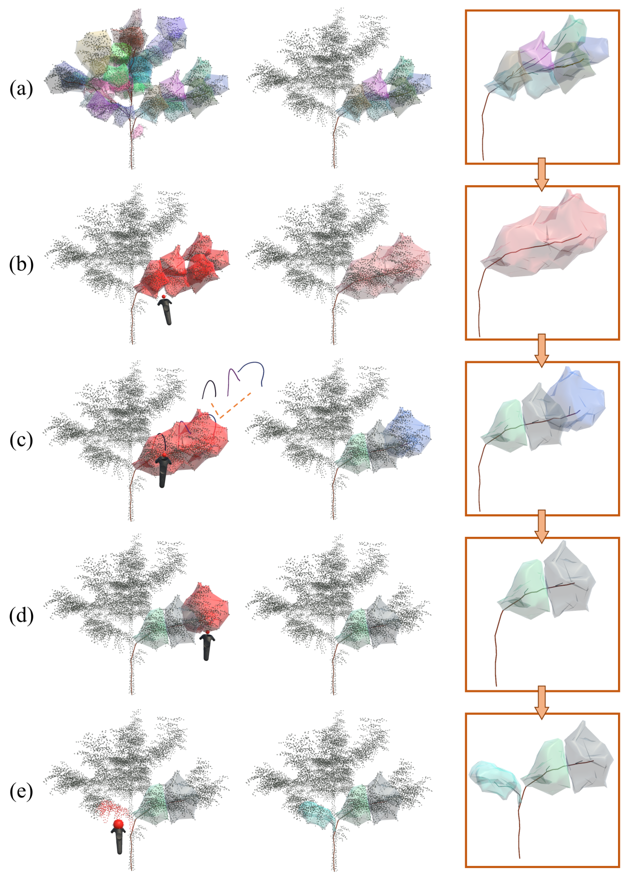


Figure 5: *Interactive editing. (a) Automatic decomposition result. (b) Merge lobe. (c) Split lobe. (d) Remove lobe. (e) Insert lobe.*

show/close point cloud, show one/all examples are also developed to enhance user interaction.

Examples library. Leveraging the proposed HBL decomposition and interactive refinement, HBL examples can be derived from widely collected point clouds, minimizing the effort required to construct the example library. To streamline the selection process of HBL examples and improve the system's usability, we integrate a pre-classification step that categorizes HBL examples based on the curvature of main branches and the number of lobes. As shown in Fig. 6, users can search for HBL examples using keywords like *"dense"*, *"sparse"*, *"straight"*, and *"curved"*. Examples meeting the specified criteria are prioritized in the display, effectively reducing the burden on users, especially first-time users, during the operation process.

3.5. Example-based Interactive Tree Generation

Interactive tree modeling. To assist in the interactive assembly of new HBL structures for tree modeling, we design and implement four key operations for creating and editing the tree model: *select example*, *transform example*, *transform lobe*, *sketch new example*. In this way, our system allows the user to interact with entire

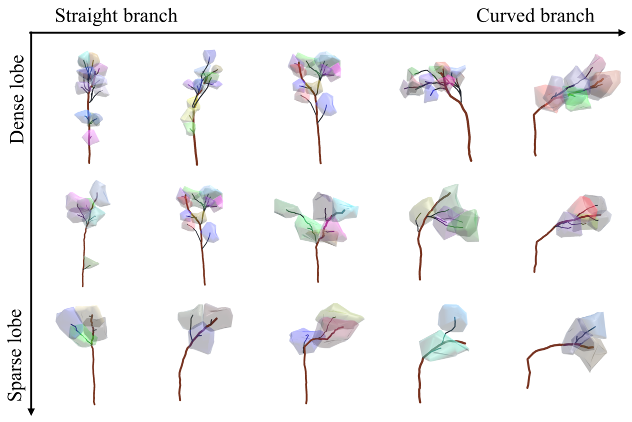


Figure 6: *Branch curvature and lobe number of HBL examples in our library.*

examples or individual lobes at different levels, making it easy to create and edit tree models. Note that each example consists of a main branch with several lobes, each connected to the main branch through secondary branches.

- **Select example**: select the needed examples.
- **Transform example**: translate, rotate and scale examples.
- **Transform lobe**: translate, rotate and scale the lobes.
- **Sketch new example**: create the free-form examples.

Based on the specified example library, users can directly interact with the HBL intermediary to synthesize tree models according to their intentions, without worrying about local botanical fidelity. As illustrated in Fig. 7, the interactive modeling process is smooth and intuitive. When users enter the system interface (a), a selection of pre-generated examples is available. Users can transform branches and lobes until they are satisfied with the layout (b-c). To avoid the need for precise interactions in VR, we provide a recombination method that allows users to arrange HBL examples in a coarse placement. Specifically, we compute the similarity of main branches between two different HBL examples. When a user-defined requirement is satisfied, as outlined in Algorithm 1, our system automatically merges and optimizes these main and secondary branches, then connects the two examples into a cohesive whole. If there is no suitable example available in the library (d), users can also sketch a few strokes to create a new example, where VRTree recognizes strokes that represent branches or lobes, similar to mid-air sketching proposed by Zhang *et al.* [ZLC*21]. Through easily selecting, transforming, and sketching examples, users can effectively generate the newly assembled HBL structure encapsulating a tree model (e).

Procedural growth from HBL. Once all examples selected by the user are positioned properly, a specially devised Space Colonization algorithm [RLP07] is employed to generate detailed twigs and leaves. For each lobe, the terminal nodes of the corresponding secondary branch are designated as the bud nodes. The growth direction $\vec{V}$ is determined by a combination of the heading direction $\vec{V}_s$, the attraction direction $\vec{V}_{att}$, and the gravity direction $\vec{V}_g$ as

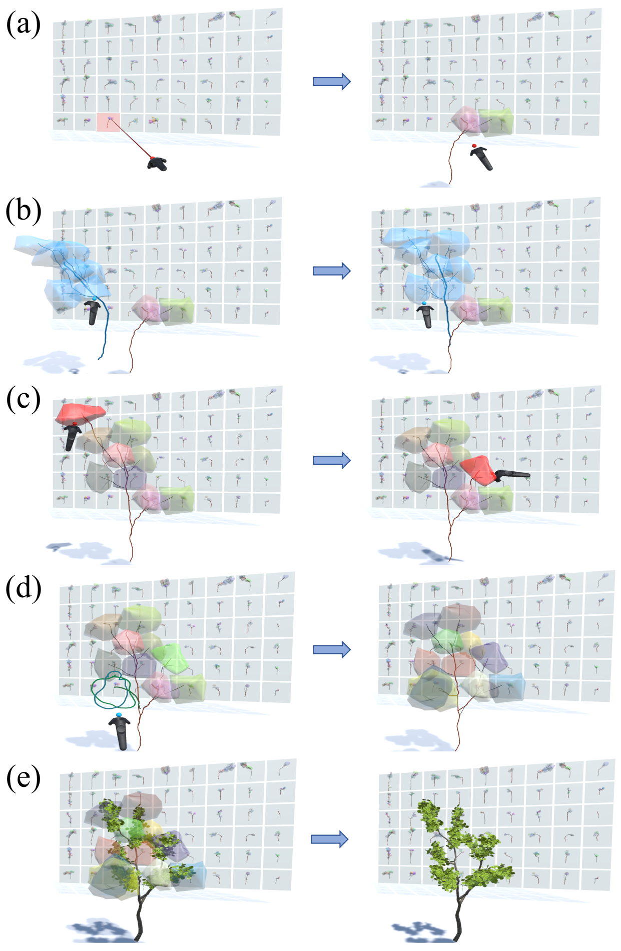


Figure 7: *Interactive modeling. (a) Select example. (b) Transform example. (c) Transform lobe. (d) Sketch a new example. (e) Twigs and leaves generation.*

follows:

$$\vec{V} = \vec{V}_s + \alpha\vec{V}_g + \beta\vec{V}_{att}. \quad (2)$$

Here, the heading direction $\vec{V}_s$ for each lobe is the tangent direction of the secondary branch. The gravity influence, represented by $\vec{V}_g = (0, -1, 0)^T$ affects the downward growth of twigs. The attraction direction $\vec{V}_{att}$ is determined by averaging the directions from the bud to each vertex of the lobe geometry, where these vertices are regarded as attraction points. The typical values of α and β are set to 1.6 and 0.3, respectively, as suggested by Guo *et al.* [GXY*18]. New nodes are generated incrementally based on a growth radius. To create a complete tree model, branch geometry is generated by fitting generalized cylinders to the corresponding point clouds with a gradually decreasing radius. Finally, leaves and textures are added around the new nodes using the method described by Livny *et al.* [LYO*10]. In practice, users can control the growth process via the controllers, allowing them to adjust the density of twigs and leaves according to their preferences.

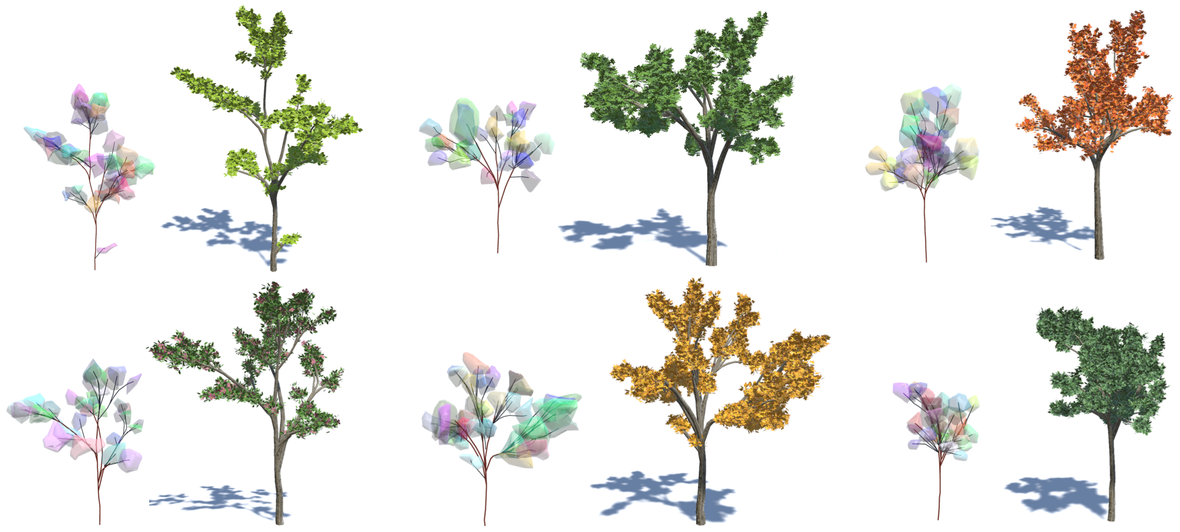


Figure 8: *A gallery of tree models created in freestyle. These 3D tree models are created by inexperienced users within 2-5 minutes using a consumer-level VR device.*

4. Results and Evaluation

In this section, we begin by introducing the implementation details in (Section. 4.1). Subsequently, we evaluate our framework on the robustness of modeling capability in three aspects: freestyle tree modeling, tree editing and image-guided tree modeling (Section. 4.2). After that, we compare our work with other state-of-the-art tree modeling approaches (Section. 4.3):

- Du *et al.* [DLL*19] and Guo *et al.* [GXY*18] are two novel baselines for automatic tree reconstruction from point clouds. We compare our method with theirs on tree reconstruction in extreme cases (point clouds incompleteness).
- Zhang *et al.* [ZLC*21] introduce a similar interactive tree modeling system in VR environments akin to us, whereas they harness the 3D sketch-based tree modeling ideology with texture-lobes as an intermediary. We compare our tree modeling performance with theirs.

Finally, a comprehensive user study is conducted to quantitatively evaluate our VRTree system and compare it with the approach proposed by Zhang *et al.* [ZLC*21] (Section. 4.4).

4.1. Experimental Setup

Implementation details. Our system is designed as server-client mode. The server is programmed by Python3.7, which is responsible for receiving messages from the client and conducting function scripts, then storing processed data in a sharing disk. The client is implemented in Unity2019.4.22f1&C#, which is activated to download and render tree-related data such as point clouds, lobe geometry and skeletal structure. In addition, we adopt the lightweight messaging library ZeroMQ to transport and parse messages. When the back-end server starts up, the client will send a message after finishing interactions (mentioned in Section. 3), receive a response message after the server's computation, then refresh the rendering for the user. We implement our system on a PC with Intel i9-12900, 2.40GHz CPU, 32G memory, and a 64-bit Windows 10 operation system. The GPU used for rendering and computation is NVIDIA RTX 3080 GPU. To build the Multi-level Tree Examples Library (MTEL), we collect a point cloud dataset based on SmartTree [DBGA23], comprising 6 distinct tree species.

4.2. Evaluation

Freestyle generation. For virtual creation enthusiasts who may not have strict requirements of modeling goals, HBL examples can serve as fundamental components, providing assistance to users. Therefore, we initially perform experiments involving free-form creation, allowing users to create trees of various species according to their imagination. In particular, users select HBL examples from the MTEL based on their intent and arrange them according to their preferences. Fig. 8 displays part of the generated results. The experiment illustrates that HBL examples integrate sufficient botanical features to guide and assist users in the interactive modeling process, significantly enhancing the user experience.

Tree editing. Innovatively, we highlight the significant advantage of our interactive system through a tree editing task that is commonly considered challenging for conventional iterative tree modeling approaches [Lin68, Pru86, SJJ09, QQJ11, PL12]. To be specific, we conduct tree editing by augmenting a pre-generated 3D tree model with an additional HBL example or substituting part of the current tree model with novel local structures. We accomplish this by first positioning additional HBL examples at the user-specified locations. Then, applying the proposed Algorithm 1 on corresponding parts and procedural growth, novel trees are substantiated by seamlessly incorporating the new HBL example while preserving the original pre-generated tree. As illustrated in Fig. 9, utilizing the 3D interactive editing, we are able to generate variations of tree species based on one individual tree model. This char-

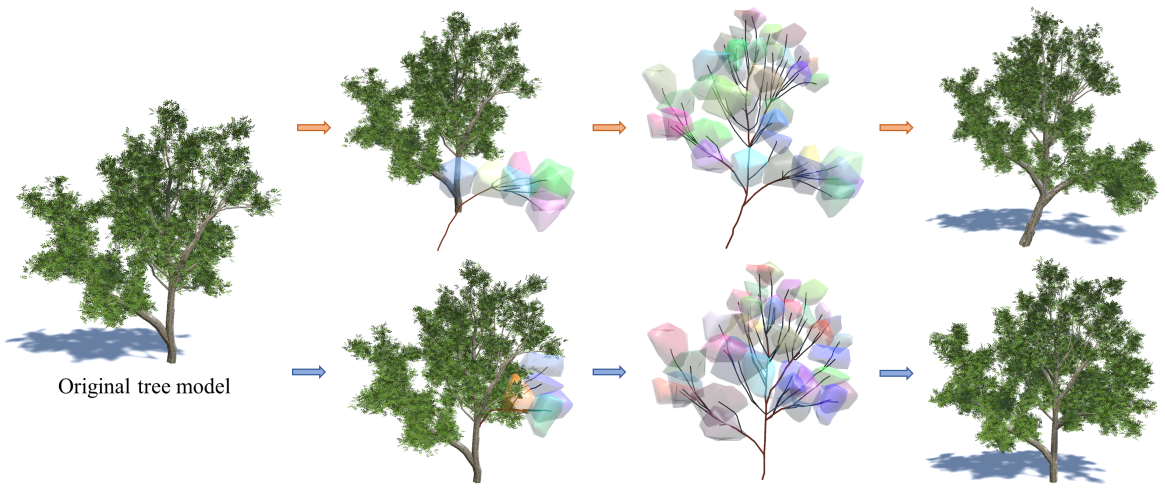


Figure 9: *Editing an existing tree model through adding novel HBL examples to generate variations of tree species, each within 1 minute.*

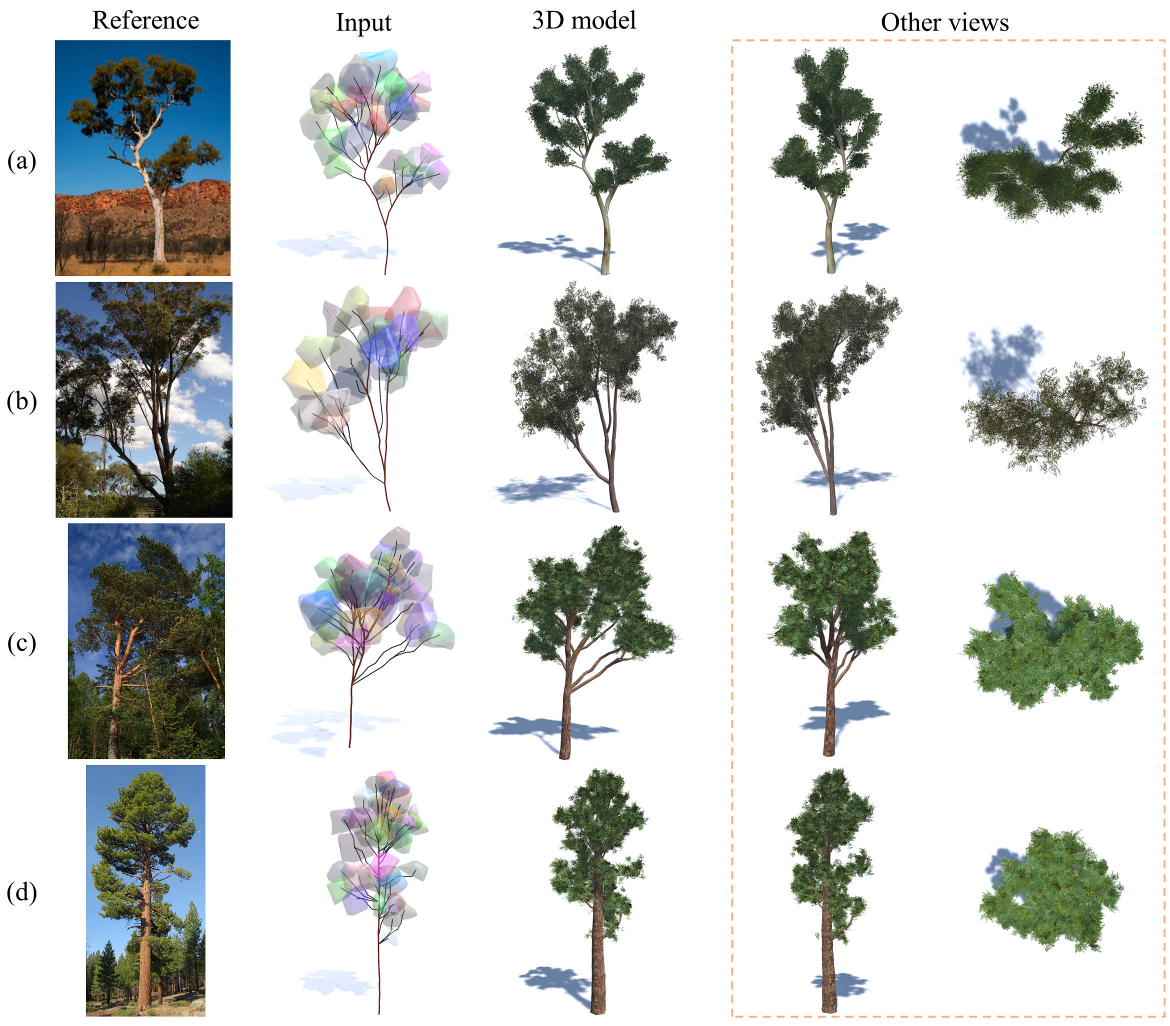


Figure 10: *Synthesized 3D tree models of various species (eucalyptus (a-b), pine (c-d)) based on our system using reference images.*

acteristic of our system endows users with enhanced flexibility in creating tree models, allowing continual editing until satisfactory results are achieved.

Image-guided tree modeling. The objective of image-guided tree modeling is to produce a 3D tree model in which one projection view closely matches the reference image. The challenge mainly lies in the absence of depth information and the potential occlusion in 2D images, impeding users from crafting tree models that accurately reflect both fidelity to the image and the desired level of realism. To evaluate our framework on this formidable topic, we first obtain photographs of two eucalyptus trees and two pine trees from the internet, using them as the reference for modeling. Next, we pre-generate the Multi-level Tree Examples Library (MTEL) according to corresponding tree species. In our experiment, we decompose point clouds of eucalyptus and pine trees to construct two sub-libraries, preserving the consistency between the source of HBL examples and the target of tree models. Finally, we select and assemble HBL examples from the MTEL to generate trees according to the reference images, and this process is achieved interactively. As shown in Fig. 10, we present the front, side, and top views of our generated tree model, along with the corresponding HBL structure. Experimental results demonstrate the capability of our system to generate tree models with high fidelity toward the reference image. Moreover, with the aid of the constructed HBL examples library, our results exhibit faithful details of trees and achieve high realism when being projected from other viewpoints.

4.3. Comparisons

Tree reconstruction from incomplete point clouds. Tree reconstruction from incomplete point clouds is a frequently encountered yet under-researched topic. The lack of overall shape may potentially mislead the growth of tree branches, yielding trees with a low level of realism. Two state-of-the-art works: Guo *et al.* [GXY*18] propose a pointcloud-guided space colonization method for branch growing, while Du *et al.* [DLL*19] adopt a heuristic directional-field optimization algorithm based on the point cloud. Both approaches demonstrate inferior performance when handling incomplete data, as depicted in (a) and (b) of Fig. 11. On the contrary, drawing upon our example-based modeling philosophy, we are capable of exploiting the framework into individual trees and addressing this issue in an interactive manner. Initially, we decompose the HBL examples from the existing portion of the point cloud, with which we construct a small HBL example library. The preliminary reconstruction results are shown in (c) of Fig. 11. Subsequently, we fill the incomplete tree by leveraging HBL-level interaction (introduced in Section 3.5) on the small library until a satisfactory appearance is attained. Fig. 11 showcases the superiority of our reconstruction results and further spotlights the generalizability of our framework to versatile tree-related scenarios. (as demonstrated in (d) of Fig. 11). This experiment demonstrates a potential extension of our method to assist in improving the reconstruction of trees from point clouds, where our system can leverage human knowledge and observation as prior information to fill in the unscanned areas by using the extracted HBL structures.

3D sketching-based generation. As illustrated in Fig. 12, we conduct a comparative analysis with another related VR interactive tree

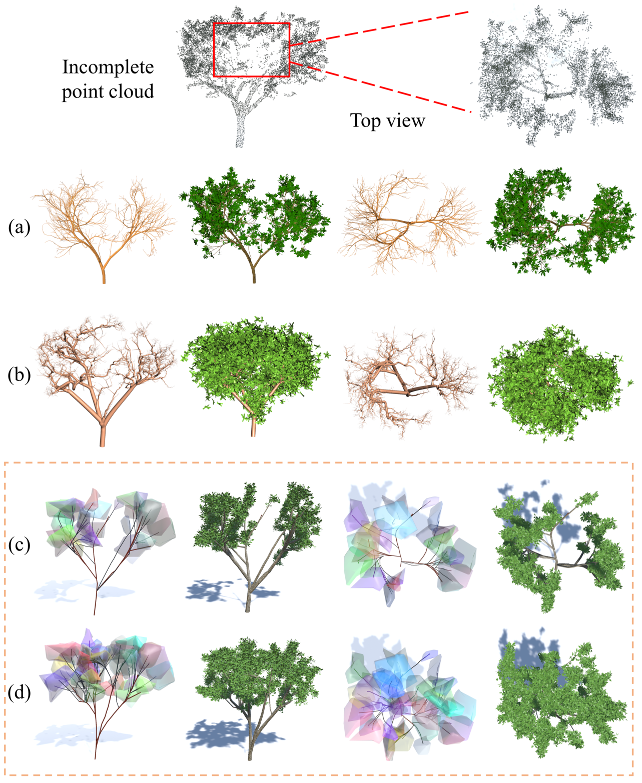


Figure 11: *Tree reconstruction based on the incomplete point cloud. Two state-of-the-art pointcloud-based tree reconstruction approaches, namely Guo et al. [GXY*18] (a) and Du et al. [DLL*19] (b), struggle to accurately reconstruct trees. Though our approach achieves similar reconstruction quality as theirs initially (c), benefiting from the interactive completion of the HBL, our reconstruction surpasses theirs significantly, attaining a tree model with a higher-level realism (d).*

modeling system [ZLC*21]. Instead of utilizing the example-based modeling framework, they apply a complete sketching paradigm. Users are tasked not only with drawing branches, which is relatively easier, but also with creating complex lobe geometry from scratch. This prevents users from drawing complicated lobe geometries where twigs grow, thereby generating unrealistic tree crowns comprising ellipsoidal-like lobes (as shown (a) of Fig. 12). Conversely, our system, featuring multi-level control and more precise HBL manipulation, synthesizes trees with a higher-level realism (as outlined in (b) of Fig. 12).

4.4. User Study

We conduct a user study and collect feedback to compare our system with Zhang *et al.* [ZLC*21]'s system. Previously, modeling 3D trees is an extremely complicated task that requires expert knowledge of CAD software. Thus, our initial intention is to demonstrate that our system can empower even novice users to handle this task easily. We recruit 8 users from local universities who are unfamiliar with tree modeling in VR environments but are highly interested,

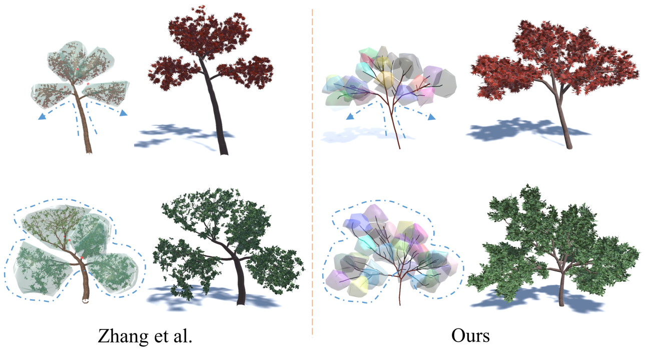


Figure 12: *Comparison with Zhang et al. [ZLC*21]'s system on tree modeling. To enable fair comparison, we assemble the HBL examples according to the coarse silhouette of the tree generated by Zhang et al. [ZLC*21]'s system. Thanks to the proposed framework, our tree modeling results demonstrate enhanced realism, reflected in more faithful botanical structure and vivid tree crown details.*

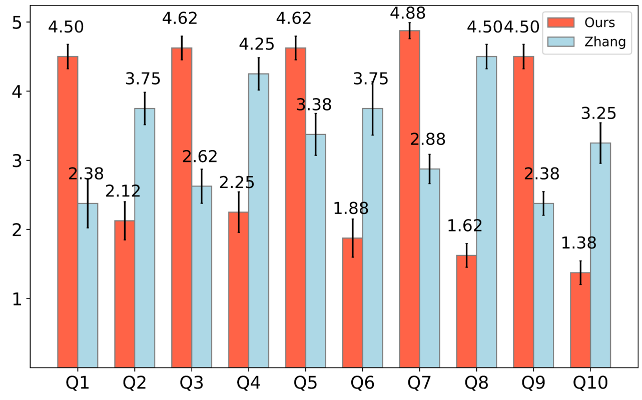


Figure 13: *The plot results of SUS questions.*

Table 1: *The questions of the System Usability Scale (SUS).*

SUS Question Description	
(Q1)	I'd like to use it frequently (if modeling trees).
(Q2)	The system is unnecessarily complex.
(Q3)	The system is easy to use.
(Q4)	I need support of a technical person.
(Q5)	The functions are well integrated.
(Q6)	This system has too much inconsistency.
(Q7)	The system is easy to learn for most people.
(Q8)	I found the system very cumbersome to use.
(Q9)	I felt very confident using the system.
(Q10)	I needed to learn lots of things before I could get going with this system.

to model trees using the two systems. The group consists of four females and four males, with an average age of 26.7 years. At the beginning, all participants are instructed to watch demonstration videos showcasing tree modeling using our system and Zhang *et al.* [ZLC*21]'s system respectively. Each video lasts 15 minutes, providing instructions on VR equipment and guidance on tree modeling. Then, participants are asked to use the two systems for tree modeling respectively. During the study, users experience the different systems in a randomized order. The first task is open-ended, allowing them to model trees freely. The second task involves providing a reference image as a target, requiring participants to create a model as similar as possible to the image. Lastly, they answer the questionnaire to evaluate the effectiveness and convenience of the two systems. The questionnaire is the System Usability Scale (SUS) [B*96] which contains ten questions on a 5-point scale. Table 1 shows the detailed SUS question description. Afterward, we carry out a brief interview with each participant to gather their feedback and experiences.

We use bar plots to compare our system with Zhang *et al.* [ZLC*21]'s system in terms of average scores. As shown in Fig. 13, all participants show a positive attitude to our system. The results show our system is simple $(Q2, M = 2.12, SEM = 0.30)$ and user-friendly $(Q3, M = 4.62, SEM = 0.18)$, while Zhang *et al.* [ZLC*21]'s system is hard to operate in 3D space due to the instability of hand-drawn sketches $(Q3, M = 2.62, SEM = 0.26)$. Participants often have to repaint when using their system $(Q6, M = 3.75, SEM = 0.41)$. For our system, users can model trees directly without requiring extensive botanical knowledge $(Q9, M = 4.50, SEM = 0.19)$. Besides, our system integrates various editing tools to facilitate users in creating tree models of specified shapes $(Q5, M = 4.62, SEM = 0.18)$, without adding any burden to the modeling process $(Q8, M = 1.62, SEM = 0.18)$. However, Zhang *et al.* [ZLC*21]'s system requires drawing a few lobes, which significantly affects the efficiency of modeling. For example, one participant commented:"*I found it easy to obtain the desired model quickly by assembling the tree examples in VRTree, compared to spending a lot of time drawing lobes and branches in another system.*"

5. Conclusion and Future Work

In this paper, we design a novel 3D interactive modeling system in VR environments using an example-based generation framework. We introduce a multi-level tree representation, HBL (Hierarchical Branch Lobe), as an intermediary to decompose tree point clouds into a series of concise HBL examples. Using this intuitive system, we construct a Multi-level Tree Examples Library (MTEL) containing HBL examples from 6 distinct tree species. Extensive experiments and user studies demonstrate the system's effectiveness and ease of use. Our system can generate 3D tree models with realistic species characteristics based on various user prompts (free-form positioning, images, point clouds) and can handle challenging tasks such as modeling incomplete tree point clouds or editing existing trees. However, our system has some limitations. One major limitation is that the manipulation of HBL examples is relatively free-form, which can occasionally result in artifacts in the generated models. In the future, we plan to address this issue by developing a recommendation system based on the current modeling state to provide guidance to users.

Acknowledgements

This work was supported by National Key R&D Program of China (2022ZD0160801), Guangdong Major Project of Basic Research (2023B0303000016), NSFC (U21A20515, 62377004), and Shenzhen Science and Technology Program (GJHZ20210705141402008).

References

[AAC20] ARGUDO O., ANDÚJAR C., CHICA A.: Image-based tree variations. In *Comp. Graph. Forum* (2020), vol. 39, pp. 174–184. 3

[B*96] BROOKE J., ET AL.: Sus-a quick and dirty usability scale. *Usability evaluation in industry 189*, 194 (1996), 4–7. 11

[BMP02] BELONGIE S., MALIK J., PUZICHA J.: Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 4 (2002), 509–522. 5

[CDOFJ22] CÁRDENAS-DONOSO J. L., OGAYAR C. J., FEITO F. R., JURADO J. M.: Modeling of the 3d tree skeleton using real-world data: A survey. *IEEE Trans. on Vis. and Comp. Graph.* (2022). 2

[CKGK11] CHAUDHURI S., KALOGERAKIS E., GUIBAS L., KOLTUN V.: Probabilistic reasoning for assembly-based 3d modeling. In *ACM Trans. Graph.* 2011, pp. 1–10. 2

[CNX*08] CHEN X., NEUBERT B., XU Y.-Q., DEUSSEN O., KANG S. B.: Sketch-based tree modeling using markov random field. In *ACM Trans. Graph.* 2008, pp. 1–9. 3

[CRW*20] CHAUDHURI S., RITCHIE D., WU J., XU K., ZHANG H.: Learning generative models of 3d structures. In *Comp. Graph. Forum* (2020), vol. 39, pp. 643–666. 2

[DBGA23] DOBBS H., BATCHELOR O., GREEN R., ATLAS J.: Smart-tree: Neural medial axis approximation of point clouds for 3d tree skeletonization. In *Iberian Conference on Pattern Recognition and Image Analysis* (2023), pp. 351–362. 3, 8, 13

[DLL*19] DU S., LINDENBERGH R., LEDOUX H., STOTER J., NAN L.: Adtree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing 11*, 18 (2019), 2074. 2, 3, 4, 5, 8, 10

[DXA*19] DUBROVINA A., XIA F., ACHLIOPTAS P., SHALAH M., GROSCOT R., GUIBAS L. J.: Composite shape modeling via latent space factorization. In *IEEE International Conference on Computer Vision (ICCV)* (2019), pp. 8140–8149. 2

[FXX*22] FU Z., XU R., XIN S., CHEN S., TU C., YANG C., LU L.: Easyvrmodeling: Easily create 3d models by an immersive vr system. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 5*, 1 (2022), 1–14. 2

[GJB*20] GUO J., JIANG H., BENES B., DEUSSEN O., ZHANG X., LISCHINSKI D., HUANG H.: Inverse procedural modeling of branching structures by inferring l-systems. *ACM Transactions on Graphics (TOG) 39*, 5 (2020), 1–13. 6

[GXY*18] GUO J., XU S., YAN D.-M., CHENG Z., JAEGER M., ZHANG X.: Realistic procedural plant modeling from multiple view images. *IEEE transactions on visualization and computer graphics 26*, 2 (2018), 1372–1384. 2, 4, 5, 7, 8, 10

[HJA20] HO J., JAIN A., ABBEEL P.: Denoising diffusion probabilistic models. *Advances in neural information processing systems 33* (2020), 6840–6851. 3

[IOI06a] IJIRI T., OWADA S., IGARASHI T.: Seamless integration of initial sketching and subsequent detail editing in flower modeling. In *Computer Graphics Forum* (2006), vol. 25, pp. 617–624. 2

[IOI06b] IJIRI T., OWADA S., IGARASHI T.: The sketch l-system: Global control of tree modeling using free-form strokes. In *International symposium on smart graphics* (2006), pp. 138–146. 2

[JZF*21] JIANG Y., ZHANG C., FU H., CANNAVÒ A., LAMBERTI F., LAU H. Y., WANG W.: Handpainter-3d sketching in vr with hand-based physical proxy. In *Proceedings of the 2021 CHI conference on human factors in computing systems* (2021), pp. 1–13. 2

[KCKK12] KALOGERAKIS E., CHAUDHURI S., KOLLER D., KOLTUN V.: A probabilistic model for component-based shape synthesis. *ACM Trans. Graph. 31*, 4 (2012), 1–11. 2

[KYC*17] KRS V., YUMER E., CARR N., BENES B., MĚCH R.: Skippy: Single view 3d curve interactive modeling. *ACM Trans. Graph. 36*, 4 (2017), 1–12. 2

[LGB*21] LIU Y., GUO J., BENES B., DEUSSEN O., ZHANG X., HUANG H.: Treepartnet: Neural decomposition of point clouds for 3d tree reconstruction. *ACM Trans. Graph. 40*, 6 (2021), 232:1–232:16. 2, 3

[Lin68] LINDENMAYER A.: Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of theoretical biology 18*, 3 (1968), 280–299. 2, 3, 8

[LKK*21] LI B., KAŁUŻNY J., KLEIN J., MICHELS D. L., PAŁUBICKI W., BENES B., PIRK S.: Learning to reconstruct botanical trees from single images. *ACM Trans. Graph. 40*, 6 (2021), 1–15. 2, 3

[LLB23] LEE J. J., LI B., BENES B.: Latent l-systems: Transformer-based tree generator. *ACM Trans. Graph. 43*, 1 (2023), 1–16. 3

[LLB*24] LI Y., LIU Z., BENES B., ZHANG X., GUO J.: Svdtree: Semantic voxel diffusion for single image tree reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024). 2

[LLT*24] LIU Z., LI Y., TU F., ZHANG R., YOKOYA N., CHENG Z.: Deeptreesketch: Neural graph prediction for faithful 3d tree modeling from sketches. In *ACM CHI conference on Human Factors in Computing Systems (CHI)* (2024). 3

[LPC*11] LIVNY Y., PIRK S., CHENG Z., YAN F., DEUSSEN O., COHEN-OR D., CHEN B.: Texture-lobes for tree modelling. *ACM Trans. Graph. 30*, 4 (2011), 1–10. 2, 3, 4

[LRBP12] LONGAY S., RUNIONS A., BOUDON F., PRUSINKIEWICZ P.: Treesketch: Interactive procedural modeling of trees on a tablet. In *SBIM@ Expressive* (2012), pp. 107–120. 3

[LSL*19] LIU Z., SHEN C., LI Z., WENG T., DEUSSEN O., CHENG Z., WANG D.: Interactive modeling of trees using vr devices. In *2019 International Conference on Virtual Reality and Visualization (ICVRV)* (2019), pp. 69–75. 1, 3

[LWG*21] LIU Z., WU K., GUO J., WANG Y., DEUSSEN O., CHENG Z.: Single image tree reconstruction via adversarial network. *Graphical Models* (2021). 2

[LYO*10] LIVNY Y., YAN F., OLSON M., CHEN B., ZHANG H., EL-SANA J.: Automatic reconstruction of tree skeletal structures from point clouds. In *ACM Trans. Graph.* 2010, pp. 1–8. 2, 3, 4, 7

[LZC21] LIU Z., ZHANG F., CHENG Z.: Buildingsketch: Freehand mid-air sketching for building modeling. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (2021), pp. 329–338. 1

[LZZ*21] LUO Z., ZHOU J., ZHU H., DU D., HAN X., FU H.: Simpmodeling: Sketching implicit field to guide mesh modeling for 3d animalmorphic head design. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (2021), pp. 854–863. 2

[MCEG23] MANFREDI G., CAPECE N., ERRA U., GRUOSSO M.: Treesketchnet: From sketch to 3d tree parameters generation. *ACM Transactions on Intelligent Systems and Technology 14*, 3 (2023), 1–29. 3, 6

[NFD07] NEUBERT B., FRANKEN T., DEUSSEN O.: Approximate image-based tree-modeling using particle flows. *ACM Trans. Graph. 26*, 3 (2007), 88. 2

[NGDA*16] NISHIDA G., GARCIA-DORADO I., ALIAGA D. G., BENES B., BOUSSEAU A.: Interactive sketching of urban procedural models. *ACM Trans. Graph. 35*, 4 (2016), 1–11. 2

[Oku22] OKURA F.: 3d modeling and reconstruction of plants and trees: A cross-cutting review across computer graphics, vision, and plant phenotyping. *Breeding Science 72*, 1 (2022), 31–47. 2

[PHL*09] PALUBICKI W., HOREL K., LONGAY S., RUNIONS A., LANE B., MĚCH R., PRUSINKIEWICZ P.: Self-organizing tree models for image synthesis. *ACM Trans. Graph. 28*, 3 (2009), 1–10. 2

[PL12] PRUSINKIEWICZ P., LINDENMAYER A.: *The algorithmic beauty of plants*. 2012. 2, 3, 8

[Pru86] PRUSINKIEWICZ P.: Graphical applications of l-systems. In *Proceedings of graphics interface* (1986), vol. 86, pp. 247–253. 2, 3, 8

[QQJ11] QI H., QIU R., JIA J.: L-system based interactive and lightweight web3d tree modeling. In *Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry* (2011), pp. 589–592. 3, 8

[RLP07] RUNIONS A., LANE B., PRUSINKIEWICZ P.: Modeling trees with a space colonization algorithm. *Nph 7*, 63-70 (2007), 6. 2, 7

[SJJ09] SUN R., JIA J., JAEGER M.: Intelligent tree modeling based on l-system. In *2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design* (2009), pp. 1096–1100. 3, 8

[WBCG09] WITHER J., BOUDON F., CANI M.-P., GODIN C.: Structure from silhouettes: a new paradigm for fast sketch-based design of trees. In *Computer Graphics Forum* (2009), vol. 28, pp. 541–550. 2

[WLJ*18] WANG G., LAGA H., JIA J., XIE N., TABIA H.: Statistical modeling of the 3d geometry and topology of botanical trees. In *Comp. Graph. Forum* (2018), vol. 37, pp. 185–198. 3

[WLX*18] WANG G., LAGA H., XIE N., JIA J., TABIA H.: The shape space of 3d botanical tree models. *ACM Trans. Graph. 37*, 1 (2018), 1–18. 3

[WP95] WEBER J., PENN J.: Creation and rendering of realistic trees. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), pp. 119–128. 3

[XGC07] XU H., GOSSETT N., CHEN B.: Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Trans. Graph. 26*, 4 (2007), 19–es. 2, 4

[XYS*15] XIE K., YAN F., SHARF A., DEUSSEN O., HUANG H., CHEN B.: Tree modeling with real tree-parts examples. *IEEE Trans. on Vis. and Comp. Graph. 22*, 12 (2015), 2608–2618. 3

[YCC*20] YIN K., CHEN Z., CHAUDHURI S., FISHER M., KIM V. G., ZHANG H.: Coalesce: Component assembly by learning to synthesize connections. In *2020 International Conference on 3D Vision (3DV)* (2020), pp. 61–70. 2

[YH21] YUAN Q., HUAI Y.: Immersive sketch-based tree modeling in virtual reality. *Comp. Graph. Forum 94* (2021), 132–143. 1, 3

[YLG*18] YI L., LI H., GUO J., DEUSSEN O., ZHANG X.: Tree growth modelling constrained by growth equations. In *Comp. Graph. Forum* (2018), vol. 37, pp. 239–253. 2

[YZB*24] YUAN L., ZHIHAO L., BEDRICH B., XIAOPENG Z., GUO J.: Svdtree: Semantic voxel diffusion for 3d tree reconstruction. In *IEEE Computer Vision and Pattern Recognition (CVPR)* (2024). 3

[ZLB*23] ZHOU X., LI B., BENES B., FEI S., PIRK S.: Deeptree: Modeling trees with situated latents. *IEEE Trans. on Vis. and Comp. Graph.* (2023), 1–14. 3

[ZLC*21] ZHANG F., LIU Z., CHENG Z., DEUSSEN O., CHEN B., WANG Y.: Mid-air finger sketching for tree modeling. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)* (2021), pp. 826–834. 1, 2, 3, 7, 8, 10, 11

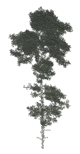
Appendix A: Tree Species and Parameter Setting.

In this appendix, we provide additional details about the tree species and parameter setting for HBL construction in our experiments.

Data preprocessing. We collect the 3D tree point clouds from Smart-Tree [DBGA23], which includes 6 tree species: *apple, cherry, eucalyptus, ginkgo, pine, walnut*. However, considering a tree with real scale and overly dense scanned points is typically not convenient for user interaction in the VR platform, the point clouds of each tree are downsampled to 12000 points and scaled to 0.8 meters beforehand.

HBL construction. When extracting shortest paths as initial branches, we empirically set the query radius to 0.05 and use $K = 5$ in the K-nearest neighbor algorithm to construct the graph. In the α-Shape algorithm for generating lobe geometry, parameter α is set to 13 by default. In Algorithm. 1 that further optimizes structures of initial branches, depending on the tree species, we extract different numbers of main branches and secondary branches, denoted as N_m and N_s, respectively. Note that since each lobe is connected to the main branches through a secondary branch, the number of lobes N_l is equal to N_s. The specific values of N_m, N_s and N_l for each tree species are shown in Table 2.

Table 2: *Our system utilizes three parameters N_m, N_s and N_l to indicate the number of main branches, the number of secondary branches, and the number of lobes, respectively.*

Apple	*Cherry*	*Eucalyptus*
$N_m = 8, N_s = N_l = 53$	$N_m = 10, N_s = N_l = 35$	$N_m = 8, N_s = N_l = 40$
Ginkgo	*Pine*	*Walnut*
		
$N_m = 8, N_s = N_l = 36$	$N_m = 6, N_s = N_l = 40$	$N_m = 10, N_s = N_l = 46$