2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Simulation-Ready Tree: High-Quality Dynamic Tree Reconstruction from a Single RGB-D Sensor

Hao Jin[1,2†], Mingxin Jiao[1†], Haoran Xie[2], Shaojun Hu[1*]

***Abstract*— Realistic animation of real trees is challenging due to the difficulty in accurately capturing and simulating their movements under varying environmental conditions. Most of real tree reconstruction methods focus on the static modeling of trees from RGB images or LiDAR point clouds. Rather than RGB images, RGB-D (RGB+Depth) sensors provide a low-cost solution for faithful reconstruction of dynamic tree models in 3D. However, it is difficult to capture and reconstruct a complete dynamic tree with complex branch structures using a single RGB-D sensor. In this paper, we propose Simulation-Ready Tree, a dynamic tree reconstruction framework that synthesizes simulation-ready trees by reconstructing 3D tree models and extracting material properties of tree branches from only a single RGB-D sensor. It starts by pre-scanning multi-view RGB-D images around an outdoor tree. For creating a complete static tree point cloud, we presented a coarse-to-fine registration method by considering the skeleton features of main branches of tree points from multi-views. Then, a static tree model is reconstructed from the registered point cloud using an improved space colonization algorithm. Subsequently, a DeT (deep RGB-D tracking) model is employed to track the movements of tree branches during pull-testing, and the material properties of the tree are approximated by Fourier transform and half-power bandwidth methods. Next, a simulation-ready tree is created by constructing its hierarchical structures with corresponding material properties. Finally, the modal analysis method of curved cantilever beams is applied to the simulation-ready tree for animating trees under static load. We demonstrate realistic animation results of our framework by comparing with the ground truth RGB-D data sequences for various tree species. The related tree animation software and demo can be accessed from the link https://github.com/treeAnimate/Simulation-Ready-Tree.**

I. Introduction

Trees add beauty to urban and forest landscapes, help absorb carbon and emit oxygen. Realistic tree simulation has extensive applications in the study of robotic tree manipulation such as fruit harvesting and tree pruning in agriculture [1], [2], and structural stability and safety assessment in forestry [3]. However, trees have complex branch structures and various species with different material properties [4], making the high-quality dynamic reconstruction of a real-world tree a challenging task.

Many previous studies have focused on reconstructing static real-world tree models from RGB images or LiDAR point clouds [5], while few of them considered the dynamic reconstruction of a specific tree in the real-world. Although Li et al. [6] and Hu et al. [7] animated plausible tree swaying from 2D videos, the 2D videos cannot reflect the real 3D motions of tree branches. With the advances of commodity RGB-D sensors, real-time 3D scanning provides the possibility for faithful dynamic scene reconstruction. The existing volumetric-based fusion methods such as DynamicFusion [8] and BundleFusion [9] works well for online reconstruction of topology-simple objects with small deformation. However, trees are deformable objects with hierarchical structure and varying mechanical properties, and these methods are not applicable to the high-quality dynamic reconstruction of trees with large deformation from a single depth sensor. Fig. 1 shows the point cloud of a tree scanned by a RGB-D camera. We can observe that the tree point cloud in the front view is much more complete than that in the side view. Using multiple depth cameras may solve this problem, however, it will increase the experiment budget and cause synchronization and dynamic registration issues. Furthermore, the generation of a simulation-ready tree is more useful than a static tree model in digital twin since it is suitable for physically-based simulation [10], which can better reflect the essential internal properties of a physical tree. Therefore, we propose a dynamic tree reconstruction framework that generates high-quality simulation-ready trees by considering their material properties from only a single low-cost RGB-D sensor.

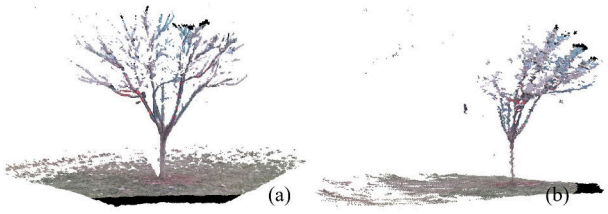

Fig. 1: The point cloud of a tree captured by a single RGB-D camera. (a) Front view; (b) Side view.

The main contributions of this paper are as follows:

- A dynamic reconstruction framework for generating a complete simulation-ready tree with complex branch structures using a single RGB-D sensor.
- The simulation-ready tree is created through the process of coarse-to-fine registration, tree geometry reconstruction, deep RGB-D tracking, material property extraction, and hierarchical structure reconstruction.
- The modal analysis method of curved cantilever beams is employed to achieve realistic tree animation for the

This work was supported in part by the Qinchuangyuan "Scientists+Engineer" team project (2025QCY-KXJ-171) and Sci-Tech project for outstanding returned overseas scientists of Shaanxi province (2025010).

[1]College of Information Engineering, Northwest A&F University, China
[2]Japan Advanced Institute of Science and Technology, Japan
[†]Equal contribution
[*]Corresponding author

simulation-ready tree.

II. Related Work

Static or dynamic reconstruction of real-world objects from depth images or point clouds is not a new research topic. However, trees have complex topologies with many detailed branches and leaves, and at the same time many of them have typical fractal patterns [11], generic surface reconstruction methods such as Poisson surface reconstruction [12] cannot be directly applied to the special objects. Therefore, a large number of reconstruction methods specifically for trees have been studied and the latest survey paper on reconstruction of static trees can be found in [5]. In this study, we focus on dynamic reconstruction of trees with large deformation from RGB-D images, thus we mainly discuss the related work of data-driven tree animation.

The earliest important studies on tree animation could date back to the 1990s in the works of [13] and [14]. Both of these works presented fundamental dynamic models for tree structures. Then, Ota et al. [15] simplified the dynamic model of branches and leaves by introducing $1/f^{\beta}$ noises, which are suitable for animating natural phenomenon. However, these methods haven't considered the motion of real-world trees. The first data-driven tree animation possibly was proposed by Diener et al. [16] in 2006. They re-targeted the 2D motion fields extracted from videos to animate 3D plant movements. In 2009, Diener et al. [17] and Habel et al. [18] employed modal analysis and tapered cantilever beams to generate real-time tree animation. Then, Li et al. [6] extended the work of [16] to generate more realistic tree animation based on a novel probabilistic model from 2D videos. In 2012, Hu et al. [19] presented a curved cantilever beam model to animate both the motion of branches and individual leaves. In 2013, Zhao and Barbič [10] presented a novel concept - "simulation-ready", which provides a convenient and ready to use data format for FEM (Finite Element Method)-based plant animation. In 2014, Pirk et al. [20] first introduced a hydrodynamics model for realistically animating the interaction between trees and a wind field. In 2017, both Wang et al. [4] and Hu et al. [7] improved the data-driven tree animation approach by considering the extraction of material properties of real trees, and Quigley et al. [21] developed a joint rigid body model for real-time tree animation. In 2018, Deul et al. [22] first applied position-based dynamics to simulate the motion of tree-like objects. Recently, Maggioli et al. [23] realistically simulated the wilting phenomenon of plants using a physically-inspired approach.

Although all the above-mentioned methods are able to generate visually realistic animation of trees. It is difficult to evaluate the realism of the generated animation because lacking of the 3D ground truth of real tree motions. Therefore, we presented a new dynamic reconstruction framework for generating a simulation-ready tree using a single RGB-D sensor. Both the static tree model and the material properties can be derived from the scanned 3D RGB-D streams, thus the realism of the generated simulation-ready tree can be evaluated with the 3D ground truth. The closest works to our study are the works of Wang et al. [4] and Hu et al. [7] because both of them have extracted material properties of trees in advance for realistic tree animation. However, they used 2D videos rather than 3D RGB-D streams as data source, and eventually the extracted material properties can only be approximate values. More recently, Lee et al. [24] generated simulation-ready trees from single images using diffusion model, however, the simulation-ready trees in [24] are designed for simulating tree development rather than for animating the deformation of trees under static or dynamic load, that is the focus of our study.

III. Methods

Fig. 2 shows our dynamic tree reconstruction framework for synthesizing realistic tree animations from a single RGB-D sensor. The whole framework includes the stages of (1) tree point clouds registration and static reconstruction, (2) tracking and estimation of material properties for branches, and (3) dynamic tree simulation. In order to generate a complete point cloud for static tree reconstruction using a single RGB-D camera, we first capture point clouds around a tree while it remains stationary (Fig. 2(a)). The tree point clouds are aligned by a coarse-to-fine registration method (Fig. 2(b)), and an improved space colonization algorithm is employed to generate a plausible static tree model from the registered points (Fig. 2(c)). At the second stage, we track the movements of tree branches from a single view during pull-testing based on a DeT model (Fig. 2(d, e)), and the material properties of branches such as natural frequencies and damping ratios can be derived from the tracked RGB-D positions (Fig. 2(f)). In the final stage, we generate a simulation-ready tree and apply a curved cantilever beam model (Fig. 2(g)) to the tree to simulate the dynamic motion of trees (Fig. 2(h)).

A. Coarse-to-fine Tree Point Cloud Registration

Preprocessing. We utilized a single consumer-grade RGB-D sensor - Kinect Azure DK camera to collect multi-view point clouds surrounding an outdoor tree. However, the Kinect camera is highly sensitive to ambient lighting in outdoor environments. Even in moderately low-light conditions, a number of outliers could appear around the tree. Moreover, the unavoidable small vibrations of the branches in nature aggravated the formation of noise on the surface of the tree point cloud as shown in Fig. 1(a). Therefore, we employed a statistical filter to remove noise and outliers according to the Euclidean distance between points. Next, a Random Sample Consensus (RANSAC) surface fitting algorithm [25] is selected to remove the ground points as shown in Fig. 3(a). In order to improve the efficiency of subsequent registration while maintaining the tree structure, we utilized the voxel grid filtering algorithm [26] to downsample the tree point cloud. In our study, the grid size was set as 2 to 3 times larger than the point cloud density, and eventually around 30% to 40% points have been removed from the input as shown in Fig. 3(b).

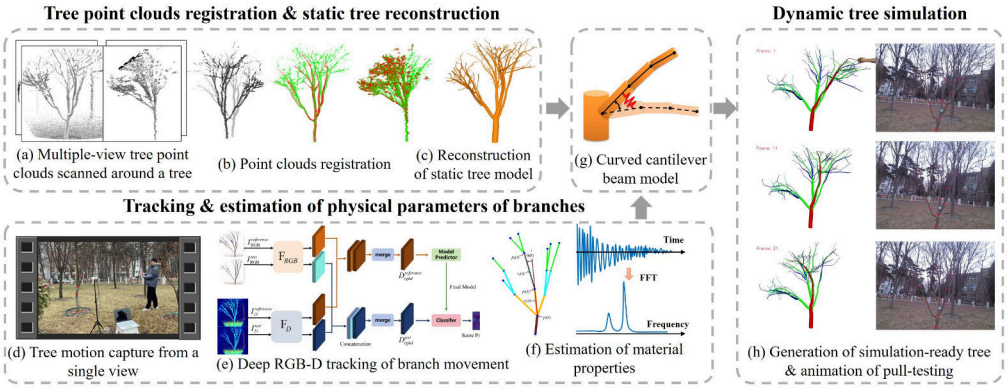

Fig. 2: The workflow of our dynamic tree reconstruction framework.

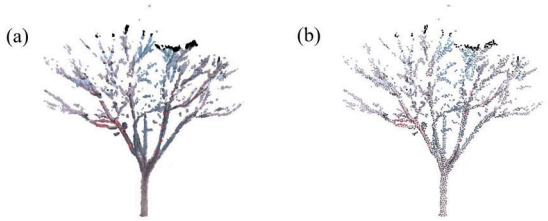

Fig. 3: Preprocessing of tree point cloud scanned from a RGB-D camera. (a) After removing noise and ground points; (b) After downsampling.

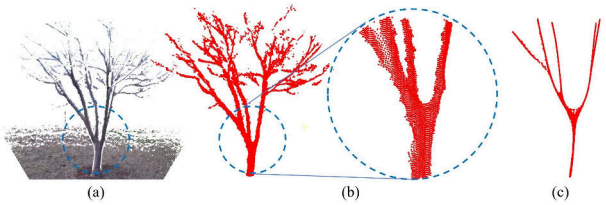

Fig. 4: Extraction of skeleton points from the selected points of main branches. (a, b) Selected featured points of main branches; (c) Extracted skeleton points based on $L1$-median method.

Registration. There are mature methods for registering point clouds of smooth and clean surfaces with a large number of overlapping areas. However, trees usually have complex topological structures with many small twigs, and for the Kinect camera with low scanning resolution, it often happens that point clouds can be collected from one view but are missing from another view, which makes it challenging to register. Therefore, we present a robust coarse-to-fine registration method by considering the invariance of skeleton points of main branches. In the coarse registration stage, we automatically select the featured points of main branches within a spherical domain centered at the branching position of the trunk with a radius as shown in Fig. 4(a, b). Next, the skeleton points are quickly extracted from the featured points using a $L1$-median-based extraction algorithm [27]. The median skeleton point is calculated by minimizing the $L1$ norm error in the region of the featured points, and the object function is denoted by:

$$argmin \sum_{i \in I} \sum_{j \in J} \|x_i - q_i\| \theta(\|x_i - q_i\|) + R(X), \quad (1)$$

where the input point cloud is $Q = (q_j)_{j \in J} \in R^3$; the projected skeleton points are $X = \{x_i\}_{i \in I}$; $\theta(r) = e^{-r^2/(h/2)^2}$ is a Gaussian function that weights nearby points; h represents the radius of local skeleton points. Function $R(X)$ constrains the distribution of the point cloud around the skeleton points to prevent excessive contraction. The final extracted tree skeleton point is shown in Fig. 4(c).

After obtaining the skeleton points, we compute the normal vector and Fast Point Feature Histograms (FPFH) for each skeleton point. Then, the Sample Consensus Initial Alignment (SAC-IA) algorithm is used to locate points in the target point cloud with similar FPFH features to those in the point cloud to be registered, and corresponding points from the source and the target can be determined. Next, we calculate the coarse transformation matrix to efficiently align the skeleton point clouds from different viewpoints into the same coordinate system.

For fine-tuning the registration of tree point clouds, we adopt the point-to-plane ICP (Iterative Closest Point) algorithm, which is suitable for point clouds that have been roughly aligned. Moreover, we additionally utilize the normal vector angle constraint for the corresponding points to eliminate incorrect pairs, thus further improving the accuracy of the registration.

B. Static Tree Reconstruction

After registering the multi-view tree point clouds, we have obtained a complete point cloud of the tree thus far. However, the subsequent dynamic reconstruction requires a complete tree model to operate. Many approaches [28]–[32] can be selected for static tree reconstruction from point clouds. In our study, we have tested AdTree [29] and the improved space colonization (SC) algorithms [32] because they have implemented the algorithms and the corresponding software is free to use. The thickness of the trunk is determined by using a M-estimator SAmple Consensus (MSAC) algorithm to fit a cylinder, and the thickness of each branch is computed according to the tree allometry formula [33]. Fig. 5 shows the

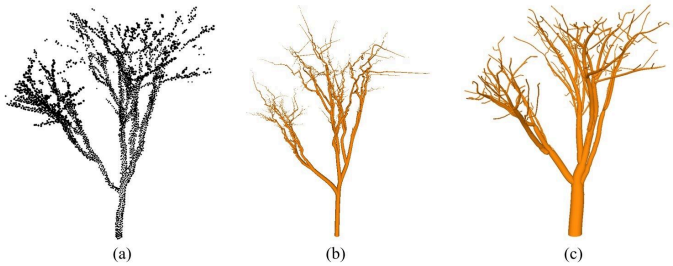

Fig. 5: Reconstruction results of a tree point with two different algorithms. (a) The registered tree point cloud; (b) Reconstructed model by the AdTree algorithm; (c) Reconstructed model by the improved SC algorithm.

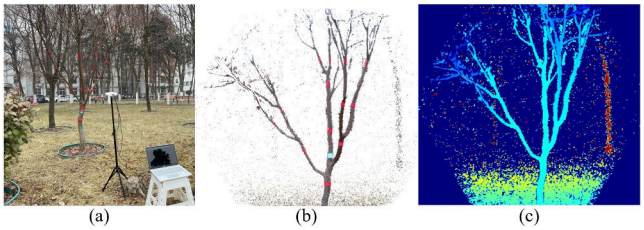

Fig. 6: Acquisition of tree RGB-D stream during pull-testing. (a) Setup of RGB-D stream capturing; (b) RGB-D point cloud from a single frame; (c) The corresponding colored depth image of (b).

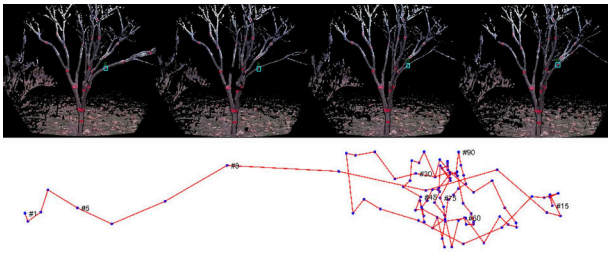

Fig. 7: The motion trajectories of a maker on a branch during pull-testing.

reconstruction results of the two algorithms. We can observe that the improved SC algorithm is able to generate smoother and natural branches than AdTree algorithm. Moreover, we can adjust the branch thickness and leaf phyllotaxis flexibly using the SC algorithm. Therefore, we select the SC algorithm for the static tree reconstruction.

C. Deep RGB-D Tracking & Material Property Estimation

Deep RGB-D tracking. We conduct a typical pull-testing, which was frequently used by arborist [34], to measure the material properties of trees. First, we collect depth image stream of branch movements after pulling and releasing a branch from a single view. Due to the self-similarity of tree branches, locating stable feature points for RGB-D tracking on the branches is a challenge. To address this problem, we wrapped red tapes around the branches as visual markers for robust tracking (Fig. 6(a)). Markers were placed based on the trees branching hierarchy. In this paper, all tree had four to six levels of branches, and markers were attached to each level of branch to ensure complete and consistent coverage. Then, the RGB-D stream can be scanned in real-time as shown in Fig. 6(b, c). Diener et al. [16] and Hu et al. [7] utilized traditional optical flow method to track the motion of branches from 2D videos. However, this method is not suitable for robust tracking the 3D movements of branches. Therefore, we employ a deep RGB-D tracking (DeT) model [35] to track different branches using depth information, which helps alleviate the problem of branch occlusion during tracking. The architecture of the DeT network is shown in Fig. 2(e).

Estimation of branch material properties. The DeT model is able to automatically track the most of the markers on the branches. For few mis-tracking markers, we manually adjust the tracking positions. Then, we can obtain the motion trajectories of these markers in 3D space as shown in Fig. 7. The result reveals that the branches undergo periodic circular vibrations after the pull and release testing. The key material properties that influence branch movements are the natural frequencies and the damping ratios [36]. Therefore, we link the markers to form a simplified tree structure as shown in Fig. 8(a). Since the movements of sub-branches include the displacements of their parent branches, we utilize the relative deflection angle $\theta_k(t)$ between the sub-branches and their parent branches over time t as the branches' real deflection signal as shown in Fig. 8(b). Finally, $\theta_k(t)$ is used as discrete signal and transformed into the frequency domain using the Fast Fourier Transform (FFT), and the natural frequency f_0 can be determined from the peak of the transformed signal in the frequency domain.

Since the motion of the branch markers shows that the branches oscillate in a periodically decaying manner when subjected to a static load, and eventually the vibrations weaken and come to a stop. The dynamic behavior of the branches shows a typical lightly damped mass spring system. Therefore, we can calculate the damping ratio ζ_0 using the half-power bandwidth method [37]:

$$\zeta_0 = \frac{f_r - f_l}{2f_0}, \tag{2}$$

where $f_r - f_l$ denotes the half-power bandwidth. The frequencies f_r and f_l are located on either side of f_0, satisfying $P_k(f_r) = P_k(f_l) = P_k(f_0)/\sqrt{2}$, where $P_k(f)$ represents the amplitude at f in the frequency domain.

D. Simulation-Ready Tree Generation

Once we have created the static tree model and extracted material properties of branches, we can represent each branch by a generalized cylinder. However, some occluded branches possibly haven't been tracked, therefore have no material properties. Considering that the single view depth images, while incomplete, covers almost all the corresponding levels from the trunk to the branches to the twigs, it is reasonable to assume that unknown parameters should be in the range of known parameters. Therefore, we approximate the material properties of occluded branches by interpolating from the

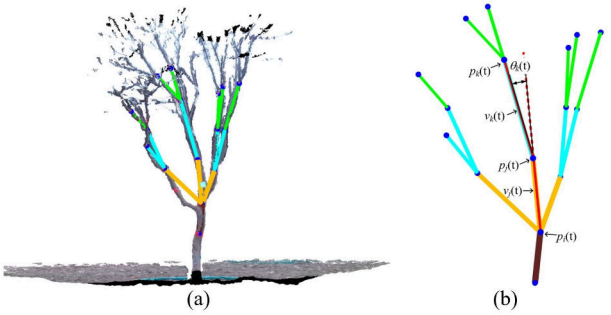

Fig. 8: The marker points on the tree are linked as a simple tree structure. (a) Tree point clouds with linked markers; (b) Illustration of the relative deflection angle $\theta_k(t)$.

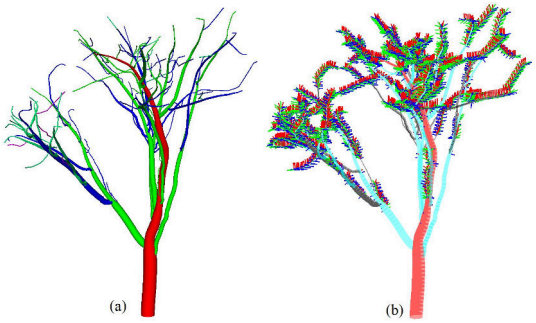

Fig. 9: (a) The hierarchy of a simulation ready tree with (b) local coordinates.

known material properties according to their lengths and thicknesses. Next, a standard depth-first-search algorithm is employed to create the hierarchy of the tree. The data structure of each branch includes material properties, local coordinates, and parent-children relations, etc. The corresponding local coordinates (u, v, w) of each branch can be computed from the directions of the current branch and its parent branch as shown in Fig. 9(b). Finally, the simulation-ready tree is generated and can be directly used for dynamic simulation.

E. Dynamic Tree Simulation

After obtaining a simulation-ready tree which is represented by a series of hierarchical generalized cylinders with material properties, we solve the dynamics of the simulation-ready tree using a modal analysis method [17]. The dynamics of the tree can be represented by:

$$\boldsymbol{M}\ddot{\boldsymbol{x}}(t) + \boldsymbol{C}\dot{\boldsymbol{x}}(t) + \boldsymbol{K}\boldsymbol{x}(t) = \boldsymbol{F}(t) \tag{3}$$

where $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$ represent the mass, damping and stiffness matrices respectively; $\boldsymbol{F}(t)$ denotes the external load. The damping matrix $\boldsymbol{C}$ is a linear combination of $\boldsymbol{M}$ and $\boldsymbol{K}$:

$$\boldsymbol{C} = \alpha\boldsymbol{M} + \beta\boldsymbol{K} \tag{4}$$

The modal analysis of an entire tree is complicated because we need to determine a large number of parameters for the matrices $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$. To simplify this process, we treat each branch as an independent mechanical system in its local coordinate system, and then directly apply Eq. 3 to each branch rather than an entire tree, that will greatly reduce the estimation of parameters and the amount of computation. Therefore, the modal analysis of an entire tree is converted to the modal analysis of a curved cantilever beam, which is used to represent an arbitrary branch. Then, the motion of a branch $\boldsymbol{X}_i(u, v, w, t)$ in its local coordinate system can be represented by the superposition of a series of modes:

$$\boldsymbol{X}_i(u, v, w, t) = \sum_{i=0}^{n-1} \boldsymbol{Y}_i(u, v, w)\boldsymbol{p}_i(t) \tag{5}$$

where $\boldsymbol{Y}_i(u, v, w)$ is mode shape and $\boldsymbol{p}_i(t)$ represents time function. In our study, we only consider the top 3 dominant modes including two translation modes along (u, w) axes and one rotation mode around v axis for simplicity. $\boldsymbol{p}_i(t)$ is represented by a damped mass spring system:

$$\ddot{\boldsymbol{p}}_i(t) + 4\pi f_i\zeta_i\dot{\boldsymbol{p}}_i(t) + 4\pi^2 f_i^2\boldsymbol{p}_i(t) = \frac{\boldsymbol{F}(t)}{m_i} \tag{6}$$

where f_i and ζ_i represent the ith natural frequency and damping ratio; m_i represents the mass of the branch. The stiffness of the ith mode is denoted by $k_i = 4m_i\pi^2 f_i^2$. Next, we substitute the extracted natural frequency and damping ratio from the tracked RGB-D data into Eq. 6 and solve $\boldsymbol{p}_i(t)$ using the classical Euler method. Finally, the $\boldsymbol{p}_i(t)$ is applied to mode shape $\boldsymbol{Y}_i(u, v, w)$ to compute the deflection of the curved cantilever beam [7].

IV. Experiments and Results

We tested 3 kinds of tree species including acer palmatum, magnolia, and acer rubrum outdoors in winter, and all the relevant data analysis and simulation were conducted on a desktop computer configured with an Intel(R) Xeon(R) Silver $2.1GHz$ CPU and a $64GB$ RAM. The depth data was captured using an Azure Kinect DK sensor with a 640×576 resolution at 30 fps. We developed an interactive dynamic tree simulation system to demonstrate the effectiveness of our method using C++ and OpenGL as shown in the github link. In the system, users can load a simulation-ready tree and flexibly pull and release the tree. Meanwhile, the user can also tune the material properties such as natural frequency, damping ratio and stiffness of each branch interactively.

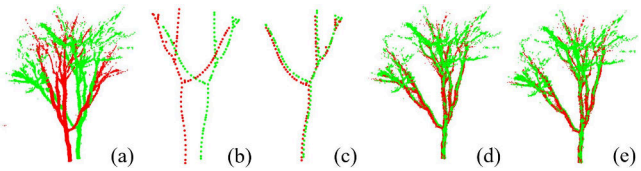

Fig. 10: Registration result of two tree point clouds with a view difference of up to 180°. (a) Initial positions of the input two point sets; (b) The extracted regional skeleton points; (c) Coarse registration of the skeleton points; (d) Coarse registration of the two tree point sets; (e) Refined registration result.

A. Evaluation of Tree Point Cloud Registration

To evaluate the robustness of our registration method, we tested an extreme registration case with two point clouds having 180° difference in viewing angles as shown in Fig. 10(a). After extracting skeleton points (Fig.10(b)), the initial transformation matrix can be efficiently computed from a small number of points. Applying the transformation matrix to the two point clouds yields a registration result shown in Fig.10(c, d). In comparison with the fine registration result shown in Fig.10(e), it can be observed that the coarse registration result based on the extracted regional skeleton points is very close to the final registration result.

To test the performance of the coarse registration, we conducted a comparative experiment on an acer rubrum. Root Mean Square Error (RMSE) was employed to assess the accuracy of alignment. Tab. I shows that our registration method by considering the skeleton features of main branches in the coarse registration stage is the most efficient one and achieved around 31% increase in successfully matched points.

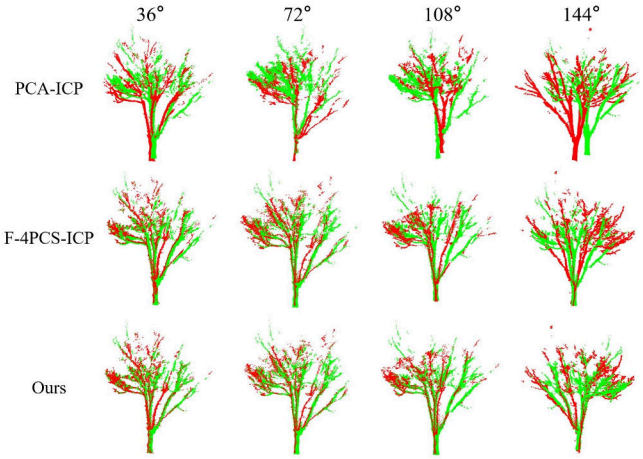

Fig. 11: Comparison with the different registration methods on acer rubrum point clouds with view differences of 36°, 72°, 108° and 144°, respectively.

TABLE I: Evaluation of coarse registration performance.

Point cloud for corase registration	Time (ms)	RMSE (mm)	Match rate
Entire tree	15744	12.44	7.08%
Key region	1759	11.59	35.08%
Tree skeleton	1623	11.88	36.36%
(Ours)Key region skeleton	**963**	**11.56**	**66.04%**

Finally, we compared our method with (1) PCA (Principal Component Analysis) coarse registration followed by an ICP precise registration (referred to as the PCA-ICP method) and (2) F-4PCS (Fast 4-Points Congruent Set) coarse registration followed by an ICP precise registration (referred to F-4PCS-ICP method) on the same acer rubrum point clouds with different view angles as shown in Fig. 11. Tab. II shows the statistics of the three registration methods for the 4 different groups of tree point clouds. When the angle interval is small, the registration accuracy of the F-4PCS-ICP method is comparable to our method. However, as the angle interval increases, our method outperforms in terms of RMSE and the number of effective point pairs. When the angle interval increases to 144°, only our method can achieve effective registration with matched points accounting for 21.59%. Additionally, the efficiency of our method has been improved around 51.3% and 39.9% in comparison with the PCA-ICP and the F-4PCS-ICP at angle intervals of 36° and 72°, respectively. The results demonstrate the robustness of our method for large interval of view angles. Furthermore, the use of a small number of skeleton points from key region in the coarse registration stage significantly improve the registration efficiency. After registration, the corresponding static tree models can be reconstructed from the relatively complete point clouds using the SC algorithm. Fig. 12 shows that the reconstructed tree models matched the registered point clouds well.

TABLE II: Quantitative comparison with the PCA-ICP and the F-4PCS-ICP.

Angle intervals	Registration method	Time (ms)	RMSE (mm)	Matched points
36°	PCA-ICP	10231	11.38	55.39%
	F-4PCS-ICP	8931	10.91	69.87%
	Ours	**4179**	**11.01**	**67.16%**
72°	PCA-ICP	12219	11.79	32.16%
	F-4PCS-ICP	9383	11.74	52.63%
	Ours	**6890**	**11.07**	**55.89%**
108°	PCA-ICP	12472	12.47	16.72%
	F-4PCS-ICP	11988	11.79	28.46%
	Ours	**7858**	**11.85**	**44.70%**
144°	PCA-ICP	17251	12.69	6.11%
	F-4PCS-ICP	12392	12.95	7.19%
	Ours	**10271**	**12.14**	**21.59%**

Fig. 12: The reconstructed tree models by using the SC algorithm. From top to bottom are acer palmatum, magnolia and acer rubrum.

B. Evaluation of Tracking & Parameter Estimation

In order to verify the effectiveness of the DeT tacking model, we compared the DeT model with the other two classical deep tracking models ATOM [38] and DiMP [39]

on our annotated tree dataset. The models were quantitatively evaluated by the standard criteria including "Success rate", "Precision", "Recall" and "F_{score}". Tab. III indicates that the DeT is better than both the ATOM and the DiMP in terms of "Success rate", "Recall" and "F_{score}", though it is a little bit lower than the DiMP in terms of "Precision".

TABLE III: Statistics of different trackers on the tested trees.

Model	Success rate	Precision	Recall	F_{score}
ATOM	0.736	0.773	0.769	0.755
DiMP	0.729	**0.809**	0.832	0.820
DeT	**0.818**	0.807	**0.851**	**0.849**

The most challenging tracking usually happens when there is occlusion and overlap between branches as shown in Fig. 13. We can observe that when the branches are overlapped, the tracked markers are occluded or multiple markers are clustered together, and the ATOM and DiMP fail to distinguish the occluded or adjacent markers, while the DeT shows its robustness in the challenging tracking test.

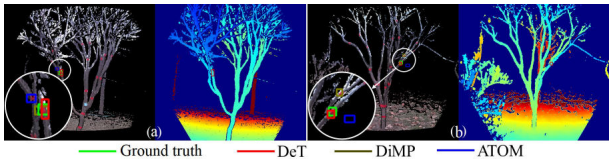

Fig. 13: Visualization results of challenging tracking with occlusion and overlap between branches.

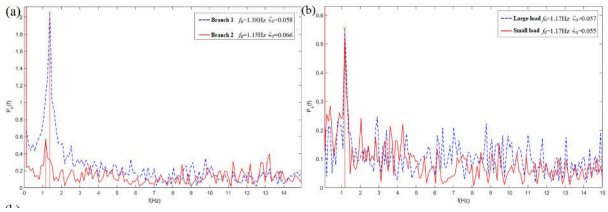

Fig. 14: Comparison of motions in frequency domain. (a) Spectra of two different branches from the same tree; (b) Spectra of the same branch under different loads.

Then, we can compute the material properties of visible branches from the tracked positions according to the method introduced in Sec. III-C. Fig. 14(a) shows the spectra of two branches from the same tree during pull-testing. We observe that different branches have different natural frequencies and damping ratios, and each branch has multiple frequency peaks, though there is only one dominant frequency peak. The spectra reveal that branches should not use the same material properties and each branch should not be represented by only one mode for realistic tree animation. Fig. 14(b) shows the spectra of the same branch under different loads. From this figure, we find that the natural frequency of the same branch doesn't change regardless of the magnitude of the external force, but the damping ratios change slightly, while the amplitude changes significantly. Therefore, we can assume that the physical parameters of a particular tree are invariant in a particular time, and the physical parameters can be assigned to a simulation-ready tree as material properties. Tab. IV shows the ranges of material properties of the tested trees. Finally, the material properties of occluded branches are interpolated from the known ranges by considering their lengths and thicknesses.

TABLE IV: Ranges of material properties of different trees.

Tree species	Natural frequency f(Hz)	Damping ratio ζ
Acer palmatum	1.07∼1.50	0.031∼0.060
Magnolia	0.72∼1.19	0.023∼0.033
Acer rubrum	0.93∼1.50	0.022∼0.056

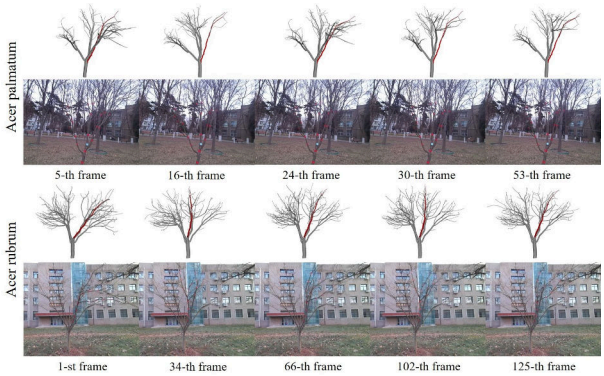

Fig. 15: Comparison of dynamic reconstruction results with real videos frame by frame.

C. Evaluation of Tree Simulation

We applied the material properties of branches to the corresponding static tree model to generate a simulation-ready tree and animated it. Next, we compared video frames from pull-testing with those from the simulation results to evaluate the realism of generated animation. Fig. 15 shows the frame by frame comparison between the simulation results and real video footage of two different trees. Through pulling and releasing a branch, the two different trees exhibit similar movements. First, the external force triggered movements across all branches, and the nearby branches showed pronounced vibrations and noticeable bending, and the swaying of branches in the simulation closely matched the motion in the real video. Finally, the branches on both the left and right sides gradually became still over time, while the pulled branch continued to oscillate slightly, showing consistent behavior between the animated and real frames. The results further demonstrate that the oscillation amplitude and motion patterns generated by our dynamic tree reconstruction framework are plausible to those in the real-world.

V. Conclusion and Future Work

In this paper, we proposed a novel dynamic tree reconstruction framework for generating a simulation-ready tree using a single RGB-D sensor. The framework has four main advantages: (1) We first used 3D RGB-D data to drive the motion of real-world trees, thus the realism of the generated

tree animation can be compared with the 3D ground truth; (2) We utilized only a low-cost single RGB-D sensor rather than multiple sensors to generate a complete tree model, and it was mainly achieved by introducing a coarse-to-fine registration method; (3) We estimated the physical properties of branches using a non-destructive way, that possibly has potential application in the tree structural stability and safety assessment for arborist; (4) The generated simulation-ready tree contains not only the geometrical information, but also the hierarchical structures and material properties, that is more suitable for the application in the field of digital twin and robotic manipulation.

Although the tree animations generated by our framework exhibit a high level of realism, due to the limited range of the sensor device, our study focused only on small trees with heights between 2 and 5 meters. In the future, we aim to explore the dynamic reconstruction of big trees with leaves in complex environments.

REFERENCES

[1] C. H. Kim, M. Lee, O. Kroemer, and G. Kantor, "Towards robotic tree manipulation: Leveraging graph representations," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 11 884–11 890.

[2] J. Jacob, T. Bandyopadhyay, J. Williams, P. Borges, and F. Ramos, "Learning to simulate tree-branch dynamics for manipulation," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1748–1755, 2024.

[3] R. B. Allison, X. Wang, and C. A. Senalik, "Methods for nondestructive testing of urban trees," *Forests*, vol. 11, no. 12, 2020.

[4] B. Wang, Y. Zhao, and J. Barbič, "Botanical materials based on biomechanics," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–13, 2017.

[5] J. L. Cardenas, C. J. Ogayar, F. R. Feito, and J. M. Jurado, "Modeling of the 3d tree skeleton using real-world data: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 12, pp. 4920–4935, dec 2023.

[6] C. Li, O. Deussen, Y.-Z. Song, P. Willis, and P. Hall, "Modeling and generating moving trees from video," *ACM Trans. Graph.*, vol. 30, no. 6, p. 112, 2011.

[7] S. Hu, Z. Zhang, H. Xie, and T. Igarashi, "Data-driven modeling and animation of outdoor trees through interactive approach," *The Visual Computer*, vol. 33, pp. 1017–1027, 2017.

[8] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 343–352.

[9] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM Trans. Graph.*, vol. 36, no. 3, 2017.

[10] Y. Zhao and J. Barbič, "Interactive authoring of simulation-ready plants," *ACM Trans. Graph.*, vol. 32, no. 4, 2013.

[11] P. Prusinkiewicz, "Graphical applications of L-systems," in *Proceedings on Graphics Interface '86/Vision Interface '86*. CAN: Canadian Information Processing Society, 1986, p. 247253.

[12] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, 2013.

[13] M. Shinya and A. Fournier, "Stochastic motion-motion under the influence of wind," *Computer Graphics Forum*, vol. 11, no. 3, pp. 119–128, 1992.

[14] J. Stam, "Stochastic dynamics: Simulating the effects of turbulence on flexible structures," *Computer Graphics Forum*, vol. 16, no. 3, pp. 159–164, 1997.

[15] S. Ota, M. Tamura, T. Fujimoto, K. Muraoka, and N. Chiba, "A hybrid method for real-time animation of trees swaying in wind fields," *The Visual Computer*, vol. 20, no. 10, pp. 613–623, 2004.

[16] J. Diener, L. Reveret, and E. Fiume, "Hierarchical retargetting of 2d motion fields to the animation of 3d plant models," in *Proceedings of the 2006 ACM Symposium on Computer Animation*, ser. SCA'06. Aire-la-Ville, Switzerland: Eurographics Association, 2006, pp. 187–195.

[17] J. Diener, M. Rodriguez, L. Baboud, and L. Reveret, "Wind projection basis for real-time animation of trees," *Computer Graphics Forum*, 2009.

[18] R. Habel, A. Kusternig, and M. Wimmer, "Physically guided animation of trees," *Computer Graphics Forum*, vol. 28, no. 2, pp. 523–532, 2009.

[19] S. Hu, N. Chiba, and D. He, "Realistic animation of interactive trees," *The Visual Computer*, vol. 28, no. 6-8, pp. 859–868, 2012.

[20] S. Pirk, T. Niese, T. Hädrich, B. Benes, and O. Deussen, "Windy trees: Computing stress response for developmental tree models," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, pp. 1–11, 2014.

[21] E. Quigley, Y. Yu, J. Huang, W. Lin, and R. Fedkiw, "Real-time interactive tree animation," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 5, pp. 1717–1727, 2017.

[22] C. Deul, T. Kugelstadt, M. Weiler, and J. Bender, "Direct position-based solver for stiff rods," *Computer Graphics Forum*, vol. 37, no. 6, pp. 313–324, 2018.

[23] F. Maggioli, J. Klein, T. Hädrich, E. Rodolà, W. Pałubicki, S. Pirk, and D. L. Michels, "A physically-inspired approach to the simulation of plant wilting," in *SIGGRAPH Asia 2023*. New York, USA: ACM, 2023.

[24] J. J. Lee, B. Li, S. Beery, J. Huang, S. Fei, R. A. Yeh, and B. Benes, "Tree-D fusion: Simulation-ready tree dataset from single images with diffusion priors," 2024.

[25] L. Li, F. Yang, H. Zhu, D. Li, Y. Li, and L. Tang, "An improved ransac for 3d point cloud plane segmentation based on normal distribution transformation cells," *Remote Sensing*, vol. 9, no. 5, p. 433, 2017.

[26] H. Guan, Y. Yu, Z. Ji, J. Li, and Q. Zhang, "Deep learning-based tree classification using mobile lidar data," *Remote Sensing Letters*, vol. 6, no. 11, pp. 864–873, 2015.

[27] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen, "L1-medial skeleton of point cloud." *ACM Trans. Graph.*, vol. 32, no. 4, pp. 65–1, 2013.

[28] A. Runions, B. Lane, and P. Prusinkiewicz, "Modeling trees with a space colonization algorithm," in *Proceedings of the Third Eurographics Conference on Natural Phenomena*, ser. NPH'07. Goslar, DEU: Eurographics Association, 2007, p. 6370.

[29] S. Du, R. Lindenbergh, H. Ledoux, J. Stoter, and L. Nan, "AdTree: Accurate, detailed, and automatic modelling of laser-scanned trees," *Remote Sensing*, vol. 11, no. 18, 2019.

[30] Y. Liu, J. Guo, B. Benes, O. Deussen, X. Zhang, and H. Huang, "TreePartNet: neural decomposition of point clouds for 3D tree reconstruction," *ACM Trans. Graph.*, vol. 40, no. 6, dec 2021.

[31] 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, no. 8, pp. 5795–5809, 2024.

[32] W. Feng, M. Jiao, N. Liu, L. Yang, Z. Zhang, and S. Hu, "Realistic reconstruction of trees from sparse images in volumetric space," *Computers & Graphics*, vol. 121, p. 103953, 2024.

[33] Y. Livny, S. Pirk, Z. Cheng, F. Yan, O. Deussen, D. Cohen-Or, and B. Chen, "Texture-lobes for tree modelling," *ACM Trans. Graph.*, vol. 30, no. 4, July 2011.

[34] F. Rinn, "Principles and challenges of static load tests ('pull-testing') for estimating uprooting safety," *Western Arborist*, pp. 36–41, 2018.

[35] S. Yan, J. Yang, J. Käpylä, F. Zheng, A. Leonardis, and J.-K. Kämäräinen, "Depthtrack: Unveiling the power of rgbd tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 725–10 733.

[36] J. R. Moore and D. A. Maguire, "Natural sway frequencies and damping ratios of trees: influence of crown structure," *Trees*, vol. 19, pp. 363–373, 2005.

[37] B. A. Olmos and J. M. Roesset, "Evaluation of the half-power bandwidth method to estimate damping in systems without real modes," *Earthquake engineering & structural dynamics*, vol. 39, no. 14, pp. 1671–1686, 2010.

[38] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Atom: Accurate tracking by overlap maximization," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4655–4664.

[39] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 6181–6190.