Learning to Reconstruct Botanical Trees from Single Images

BOSHENG LI, Purdue University
JACEK KAŁUŻNY, Adam Mickiewicz University
JONATHAN KLEIN, University of Bonn
DOMINIK L. MICHELS, KAUST
WOJTEK PAŁUBICKI, Adam Mickiewicz University
BEDRICH BENES, Purdue University
SÖREN PIRK, Google Research

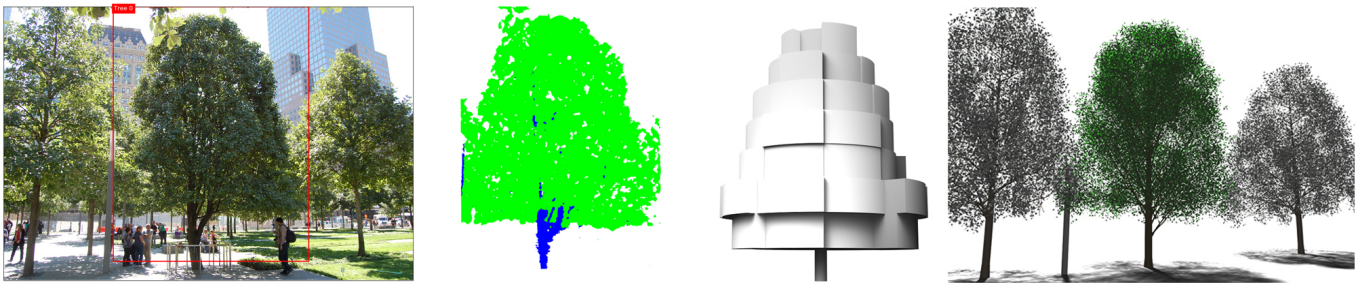

Fig. 1. Single image tree reconstruction: our method automatically reconstructs trees from single images (a). We obtain semantic segmentation masks of trees to identify branches and leaves (b). We introduce Radial Bounding Volumes (c) as a fixed-size representation for 3D tree models and show that this representation can directly be learned with neural networks. We then use the predicted RBVs to automatically reconstruct tree models with a high degree of visual fidelity (d). To reconstruct multiple trees in a single image we detect bounding boxes (a, red) before obtaining the semantic segmentation masks.

We introduce a novel method for reconstructing the 3D geometry of botanical trees from single photographs. Faithfully reconstructing a tree from single-view sensor data is a challenging and open problem because many possible 3D trees exist that fit the tree's shape observed from a single view. We address this challenge by defining a reconstruction pipeline based on three neural networks. The networks simultaneously mask out trees in input photographs, identify a tree's species, and obtain its 3D radial bounding volume – our novel 3D representation for botanical trees. Radial bounding volumes (RBV) are used to orchestrate a procedural model primed on learned parameters to grow a tree that matches the main branching structure and the overall shape of the captured tree. While the RBV allows us to faithfully reconstruct the main branching structure, we use the procedural model's morphological constraints to generate realistic branching for the tree crown. This constraints the number of solutions of tree models for a given photograph of a tree. We show that our method reconstructs various tree species even when the trees are captured in front of complex backgrounds. Moreover, although our neural networks have been trained on synthetic data with data augmentation, we show that our pipeline performs well for real tree photographs. We evaluate the reconstructed geometries with several metrics, including leaf area index and maximum radial tree distances.

CCS Concepts: • **Computing methodologies** → **Generative and developmental approaches**; **Shape analysis**; *Computer vision problems.*

Additional Key Words and Phrases: Botanical Tree Models, Tree Reconstruction, Bounding Volumes, Shape Reconstruction

ACM Reference Format:
Bosheng Li, Jacek Kałużny, Jonathan Klein, Dominik L. Michels, Wojtek Pałubicki, Bedrich Benes, and Sören Pirk. 2021. Learning to Reconstruct Botanical Trees from Single Images. *ACM Trans. Graph.* 40, 6, Article 231 (December 2021), 15 pages. https://doi.org/10.1145/3478513.3480525

1 INTRODUCTION

Due to the dominant presence of vegetation, not only in outdoor but also in urban and even indoor settings, detailed plant models significantly add to the realism of almost all virtual scenes. This spans as wide as from applications in architecture and agriculture, research in forestry and urban planning, training of autonomous agents, scene understanding for augmented reality, and content creation for games and movies. It is important to reconstruct models as faithful as possible to the captured plant's features, especially in settings where it is necessary to make informed decisions about trees or interact with vegetation. For some applications – such as autonomous driving – it is also important to frequently update the vegetation according to changes in the real world.

Authors' addresses: Bosheng Li, Purdue University, 401 North Grant Street, West Lafayette, IN, 47907-2021; Jacek Kałużny, Adam Mickiewicz University, Umultowska 87, 61-614 Poznań, Poland; Jonathan Klein, University of Bonn, Regina-Pacis-Weg 3, 53113 Bonn, Germany; Dominik L. Michels, KAUST, Visual Computing Center, Thuwal 23955, KSA; Wojtek Pałubicki, Adam Mickiewicz University, Umultowska 87, 61-614 Poznań, Poland; Bedrich Benes, Purdue University, 401 North Grant Street, West Lafayette, IN, 47907-2021; Sören Pirk, Google Research, 1600 Amphitheatre Parkway, Mountain View, CA, 94043.

0730-0301/2021/12-ART231 $15.00
https://doi.org/10.1145/3478513.3480525

Current methods for reconstructing trees from sensor data mostly rely on separately reconstructing the main branching structure and the tree crown. Central to many techniques is to obtain geometric envelopes for the tree crown, which are then populated with branches generated by a procedural model. For images, user-defined sketches [Tan et al. 2008] or image segmentation [Argudo et al. 2016] can be used to infer such 3D envelopes. Neubert et al. [2007] use two images for obtaining a 3D voxel representation to constrain the viewpoint ambiguity to then model branching structures with a particle flow algorithm. For point clouds, 3D envelopes can be generated more easily and in more nuanced ways before modeling branch geometry [Livny et al. 2011].

Reconstructing trees is challenging for many reasons: for one, the foliage often occludes parts of the main branching structure, which hinders their reconstruction. Second, even if the branches are visible, their structure is intricate and cannot always be fully captured with today's sensor hardware, such as cameras or laser scanners. Third, reconstructing a tree from a single viewpoint is under-constrained as many possible solutions exist to model the corresponding 3D tree geometry. However, it is often desirable to reconstruct a tree from only a single viewpoint despite these challenges. It is not always possible to access a tree from multiple directions. Moreover, reconstructing trees employing commodity sensor hardware, such as cameras in phones or UAVs, is advantageous compared to more precise – yet less convenient – hardware, such as laser scanners.

For many applications, such as the reconstruction of urban environments, it is necessary to generate vast amounts of 3D tree models to populate scenes realistically. However, none of the existing reconstruction methods allow reconstructing trees from single images at scale as they either require user-provided annotations for each tree or time-intensive and unintuitive parameter tuning of procedural models. To address this open research problem, we present a novel approach for reconstructing real trees from single photographs. Our pipeline is based on state-of-the-art neural networks that allow us to mask out trees from photographs, identify their species, and learn their overall geometric structure as 3D bounding volumes. We use the obtained features along with a procedural model for tree development to generate a tree model that captures the overall appearance of the captured tree. We show that the learned features enable us to prime the procedural model even to capture subtle nuances of tree form among different species. The 3D bounding volume, along with the morphologic constraints of the procedural model, enables us to faithfully generate tree geometry that is not visible in the input photographs. Furthermore, we show that our neural network pipeline can be trained solely on images of synthetically generated trees to infer parameters for real trees shown in photographs. To further showcase the usefulness of our method for content creation, we use a physics-based approach for simulating and animating tree motion. Once the tree model is generated, we convert its graph into a position-based dynamics representation and realistically simulate plant motion based on Cosserrat rods [Pirk et al. 2017].

We evaluate the success of our neural network-based tree reconstruction pipeline by quantitatively assessing the similarity of synthetic and reconstructed 3D tree models. As measuring the similarity of tree models is a challenging and open research problem, we propose a set of metrics to validate that our method works effectively, including leaf area index, maximum radial tree distances, and various tree measures applied in forestry. Additionally, we evaluate the expressiveness of our metrics by conducting a user study. We use our method to reconstruct real trees from photographs, which we qualitatively evaluate by showing a number of reconstructions. Furthermore, we compare our reconstructed tree models to results from state-of-the-art methods for reconstructing trees from photographs and laser-scanned points sets.

An example in Fig. 1 shows the capabilities of our framework. We compute a semantic segmentation mask that provides class labels for branches and leaves. We then generate a 3D structure of fixed spatial resolution from the segmentation mask that we call radial bounding volume. The mask and the bounding volume are used to control the growth of a developmental model for trees to generate a 3D model. Reconstructions of trees from real photographs are shown in Fig. 15. In summary, our contributions are: (1) we propose radial bounding volumes as a lightweight fixed-size 3D representation for tree models; (2) we propose a pipeline of three neural networks that allow us to detect and segment trees in photographs, to estimate its 3D structure based on radial bounding volumes, and to identify the tree species; (3) we introduce a novel bi-modal developmental model for trees that combines phenomenological and self-organizing growth; (4) we propose novel metrics to assess tree form similarity and apply them to evaluate the effectiveness of our learning-based reconstruction approach quantitatively; (5) in contrast to previous methods, our approach enables the automatic large-scale reconstruction of tree models from single images.

2 RELATED WORK

Research on modeling trees and plants has received a considerable amount of attention. Early approaches for modeling branching structures use rule-based algorithms [Honda 1971; Prusinkiewicz and Lindenmayer 1990], repetitive patterns [Aono and Kunii 1984; Kawaguchi 1982; Oppenheimer 1986; Smith 1984], cellular automata [Greene 1989], particle systems [Reeves and Blau 1985], or the combination of approaches [Lintermann and Deussen 1999]. As plants rarely grow in isolation, many methods focus on the plant interaction with the environment through query modules [Měch and Prusinkiewicz 1996] or random-walk [Benes and Millán 2002], inverse procedural modeling [Guo et al. 2020; Stava et al. 2014], by modeling the competition for resources and self-organization [Palubicki et al. 2009], space colonization [Runions et al. 2007], or by inversely modeling the growth response [Pirk et al. 2012b].

Recently, a number of methods started focusing on more accurately modeling the biomechanical and physical properties of plants [Pirk et al. 2016]. This ranges from modeling the interactive growth of trees [Longay et al. 2012; Pirk et al. 2012a] and climbing plants [Hädrich et al. 2017], the interaction of plants and fluids, such as wind [Habel et al. 2009; Pirk et al. 2014; Quigley et al. 2018], and fire [Pirk et al. 2017], to simulating material properties with FEM methods [Wang et al. 2017b; Zhao and Barbič 2013], and even the cambial growth of trees [Kratt et al. 2015]. These methods provide computational models and representations that enable dynamic motions of plant geometries, thereby enhancing overall realism.

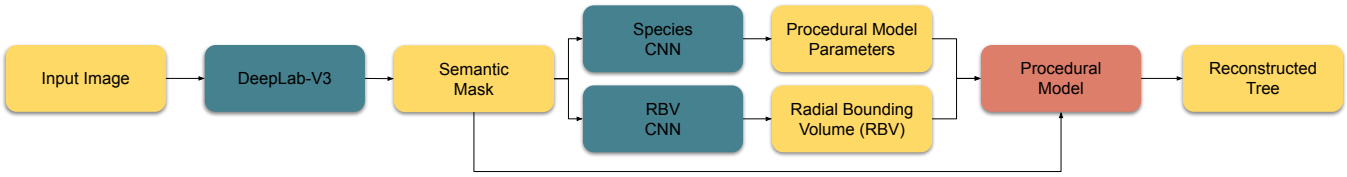

Fig. 2. Overview of our framework: we use a single image as input and employ DeepLab-V3 for semantic segmentation to obtain a semantic mask of an image that assigns each pixel with one of the three labels: background, branches, and foliage. We use the segmentation mask to train a neural network for species identification and another neural network for estimating radial bounding volumes (RBV). Based on the predicted species, we obtain parameter values for a developmental tree model. We then use the predicted semantic mask, the estimated RBV, and the selected species parameter values to compute the growth of a 3D tree model with visual traits corresponding to the input image. Colors highlight input, output, and generated data (yellow), the used neural networks (teal), and the procedural model (red).

On a different trajectory, sketch-based methods aim to realistically model branching structures while retaining artistic control. Most methods rely on user-defined sketches to model the defining features of trees [Okabe et al. 2007; Tan et al. 2008], plants [Anastacio et al. 2006], or even flowers [Ijiri et al. 2006]. User-defined sketches serve as a powerful tool that enables generating complex tree geometry. Prominent examples include converting freehand sketches through probabilistic optimization [Chen et al. 2008], employing user-defined sketches of branches as a means to guide particle flows for reconstructing tree structures [Neubert et al. 2007], or defining envelope shapes based on sketched silhouettes [Wither et al. 2009]. Moreover, it has been recognized that generating tree shapes through geometric and structural blending not only enables an efficient means of artistic control but also to obtain statistical summaries of tree populations [Wang et al. 2018a,b, 2017a].

Instead of manually modeling a plant, it has been recognized that reconstructing realistic branching structures directly from sensor data is an intriguing alternative. A number of methods aim to reconstruct tree models from different data modalities, including images [Neubert et al. 2007; Quan et al. 2006; Reche-Martinez et al. 2004; Tan et al. 2007], point clouds [Livny et al. 2011; Xu et al. 2007], or even videos [Li et al. 2011]. Due to the enormous geometric complexity of vegetation, the objective of reconstructing models of trees and plants, faithful to all their defining features, continues to be a challenging and open problem [Behrendt et al. 2005; Deussen et al. 2002; Neubert et al. 2011]. Many methods rely on point cloud data and on decomposing the reconstruction process into multiple steps or components. For example, Livny et al. [2011] explicitly reconstruct the main branches with a graph algorithm, while the foliage is only reconstructed approximately based on a set of bounding volumes. Bradley et al. [2013], on the other hand, use point clouds obtained from stereo images and learn a statistical model combined with non-rigid mesh fitting to reconstruct dense foliage. Li et al. [2013] go even further and consider the temporal domain for plant reconstruction by tracking topological events like budding and bifurcation.

Compared to point clouds, images represent a more convenient way to capture vegetation, as cameras are readily available in various devices, such as phones or UAVs. However, reconstructing trees from single-view images is difficult and an ill-posed problem. A single image does not provide enough detail to reconstruct the 3D structure of a tree meaningfully. Details that are not captured in the image cannot be easily reconstructed. To address this problem, Reche-Martinez et al. [2004] and Neubert et al. [2007] employ images of multiple views to obtain more holistic 3D representations of trees, which allow them to reconstruct their branching structure more precisely. Quan et al. [2006] and Tan et al. [2007] capture a sequence of images and use *structure from motion* to extract 3D point clouds to then reconstruct detailed models of trees and plants. Argudo et al. [2016] use image segmentation to generate 3D envelope meshes of tree crowns along with radial distance maps to render reconstructions of trees at terrain scale.

Closest to our work is the method of Tan et al. [2008] that also focuses on reconstructing trees from single images. In this work, a user needs to manually identify the main branches and the crown shape by sketching. The crown geometry is then generated from predefined branch templates, and leaves are synthesized based on the generated branching structure. Contrary to their method, we propose an automatic pipeline for reconstructing trees that does not require user input.

3 OVERVIEW

Our goal is to reconstruct realistic 3D tree models from single images efficiently and at scale which is challenging because it is ill-posed. Given the image of a tree, many possible 3D trees exist as solutions that satisfy the 2D projection. We propose a learning-based framework based on a novel representation for tree models and a two-stage reconstruction process summarized in Fig. 2. We introduce *radial bounding volumes (RBV)* for tree models that encode tree form as a set of cylindrical layers of sectors. Each sector stores the outermost spatial extent of the branching structure. We show that this definition of bounding volumes not only serves as a meaningful representation to capture complex tree form but it also can represent various species with a fixed number of layers and sectors. Moreover, RBVs can be estimated from images by deep learning.

To reconstruct trees from photographs, we first perform an image analysis based on a neural network for semantic segmentation that separates tree pixels (leaves and branching structure) from other pixels (background, other objects). We use a species identification network that extracts the species of a tree observed in a photograph, which we use to prime a developmental model for tree growth. We then use a third neural network to predict the RBV of a tree from the inferred semantic segmentation mask.

Once the RBV of a tree has been obtained, we use it, along with the semantic mask, to guide the growth process of our novel developmental tree model. We model plant development by combining a space colonization approach [Runions et al. 2007] with a phenomenological growth model for branching structures that we call *bi-modal growth model*. While space colonization provides a means to guide the branch development based on attractor points distributed in a 3D envelope, the phenomenological growth generates realistic and biologically plausible branching structures based on a number of parameters that we can obtain by predicting a species identifier with the species identification network. We then use the RBVs and the semantic mask of a captured tree to guide the placement of markers along with the main branching structure and in the 3D space of the radial bounding volume. With this setup, our framework enables the efficient and large-scale reconstruction of tree models that show the essential visual features of the captured real trees.

4 SINGLE IMAGE TREE RECONSTRUCTION

Here we introduce radial bounding volumes (RBVs) for tree models and our neural networks for performing semantic segmentation, species identification, and bounding volume estimation.

4.1 Radial Bounding Volumes

We introduce RBVs (see Fig. 3) to encode tree form in an abstract way that preserves essential shape features. An RBV is generated from an upright-oriented cylinder C with height h and radius r defined by the maximum height and maximum horizontal extent of a tree. We slice the cylinder into n layers L_i, where each layer represents a cylindrical slab of size h/n. Each layer L_i is then subdivided into a fixed number m of sectors S_{ij}. Each sector radius varies according to the amount of local occupancy. Each cylindrical sector is tightly fit to the tree geometry at this particular location resulting in a dense concave bounding volume.

Compared to common representations for tree models, such as graphs or meshes, RBVs offer the advantage of encoding tree form based on a fixed size of sectors. Compared to other fixed-size representations, such as voxels, RBVs encode tree form more efficiently by just storing the outer shape of a tree model. For example, using a lattice of size 128^3 would occupy 8,192 KB in size ($128^3 \times 4$ Bytes) using a 32-bit floating-point encoding. Instead, RBVs only require 0.25 KB ($8 \times 8 \rightarrow 64 \times 4$ Bytes). Please note, that a small voxel representation, e.g. 8^3, is not sufficient to encode important geometric features required for learning. RBVs can be defined with a fixed number of layers and sectors, and they facilitate the learning of 3D tree structure. Neural networks commonly only operate on fixed-sized input and output representations, such as images or 3D lattices. Therefore, without using RBVs, most neural network architectures could only learn to reconstruct 3D tree models based on 3D lattices. However, the resolution of 3D lattices to represent trees with their intricate structure (such as smaller branches and twigs) would need to be much higher than what can effectively be used for training neural networks. A network using 3D lattices with 128^3 resolution would need to predict ~2M values, which is orders of magnitude higher than the number of values for an RBV of resolution 8×8 resolution (64 values).

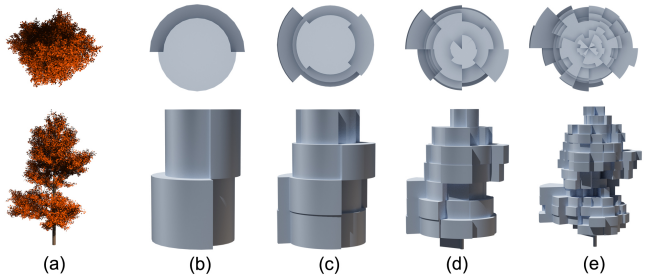

Fig. 3. Radial Bounding Volumes of a maple tree (a) with the resolutions 2×2 (b), 4×4 (c), 8×8 (d), and 16×16 (e).

Fig. 4. Examples of different data augmentation operations, including changes of contrast, brightness, and hue, as well as the transformations horizontal flip, random crop, and Gaussian blur.

4.2 Learning to Reconstruct Trees

To reconstruct tree models, our method relies on three neural networks that allow us to obtain masks of foliage and branching structure based on semantic segmentation, identify the species of a tree, and estimate the 3D RBV. In the following, we describe the network architectures and training setups for each of these networks.

4.2.1 Synthetic Dataset. There are no databases of real 3D tree models. Therefore, to train our neural networks, we only rely on synthetically generated data. Specifically, we generate a dataset with seven different tree species (acacia, maple, oak, apple, pine, willow, birch). Each tree model is generated with our developmental growth model and rendered with our framework. This enables us to jointly collect image data, such as the RGB images of rendered trees I, semantic masks of the branching structure and the foliage M, as well as geometry data such as the RBV R and surface mesh O of the tree model as well as the graph of the branching structure G. We also store the parameter values P of the developmental model that define the species of the generated tree model along with a species identifier U. Each data point in our dataset is described by the tuple $S = (I, M, R, O, G, P, U)$.

To effectively train neural networks on synthetic data to generalize on real photographs, our goal is to generate photo-realistic images of trees. To also generate a large dataset efficiently, we use a rasterization pipeline based on OpenGL. Specifically, we employ the PBR model based on Blinn-Phong illumination of the Unreal 4 engine [Karis 2013]. We use multiple light sources for smooth lighting and randomized their directions to simulate the variance of illumination that can be observed in real photographs. Additionally,

we applied different sets of albedo, normal, and roughness textures for surface material. Sectors for RBVs are defined by using a canonical camera pose. We generate RGB images by rendering each tree with a random rotation against a randomly selected background of common landscape and urban scene photographs. Semantic masks are RGB images that store three distinct color values for background (white), branches (blue), and foliage (green) and are directly obtained from the renderer. Our dataset consists of 21k individual tuples of training data, 2.7K tuples of validation data, and 2.7k tuples of test data. Examples of tree models and masks are shown in Figs. 4 and 5. It takes 14 hours to generate 26.4K data points, and the average time for generating a single instance is 1.5 seconds in our experiments. In the Appendix we show example tree models in Fig. 21 and the parameters with the values used to generate them in Tabs. 3 and 2.

4.2.2 Instance Segmentation. To reliably detect masks of plant instances along with the separation into branching structure and foliage, we rely on DeepLab-V3, a state-of-the-art network for semantic segmentation [Chen et al. 2016]. This network architecture uses atrous convolutions applied in a parallel or cascade manner to capture image features of the input images at multiple scales. This way, DeepLab-V3 robustly extracts semantic masks of tree models. Our goal is to detect masks of the main branching structure and the overall shape of the foliage of a tree to faithfully reconstruct trees from photographs. However, training the network requires ground truth data of semantic masks – pixel-precise labels of the different semantic classes – which are difficult to obtain. To the best of our knowledge, no dataset exists that provides these semantic class labels for trees. Therefore, we aim to use renderings of tree models to generate a synthetic dataset.

When a neural network is trained on synthetic data to operate on real data, the data distributions of photographs of real trees and renderings of tree models often do not match. This is referred to as *domain gap* and is caused by modeling and rendering artifacts – renderings of trees cannot be efficiently generated to the quality of photographs. To address the domain gap between data distributions, we use a data augmentation strategy based on varying the rendering configuration and applying a number of image transformations. We set the camera to a random position around the tree during rendering, adjust the number and positions of up to eight lights placed in the scene, and vary the shadow intensity. We then apply image transformations to the input image, including color, brightness, and contrast changes. Additionally, we apply a random amount of Gaussian blur, randomly apply a horizontal flip, and crop the image with a randomly sized bounding box with a minimum size of 256×256 pixels.

DeepLab-V3 is then trained on this data with the standard settings until it converges. Fig. 4 shows examples of rendered and augmented tree images, and Fig. 5 shows the ground truth and the predicted semantic masks of tree models. Please note that while there is a difference in the amount of detail between generated and predicted semantic masks, we are not interested in perfectly reconstructing the mask. Instead, our objective is to extract the main branching structure and the overall shape of the foliage, which – together – provides enough information about a tree to reconstruct it with our modeling algorithm faithfully.

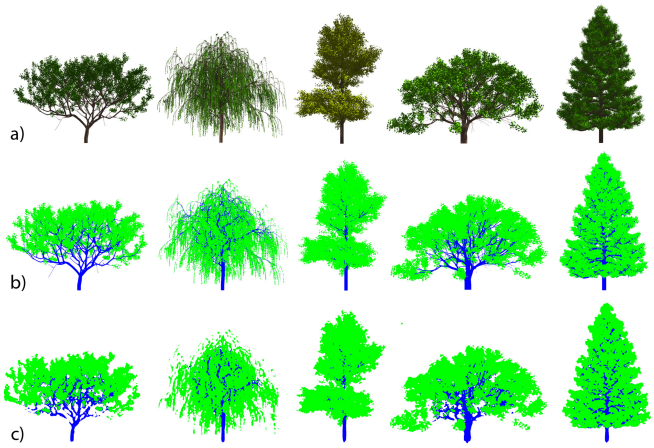

Fig. 5. Tree models and masks: we use our framework to render trees (a) and to generate semantic segmentation masks (b) that provide pixel-precise labels for branches (blue), leaves (green), and background (white). We then train a semantic segmentation network on the synthetic data to estimate segmentation masks (c).

4.2.3 Learning Radial Bounding Volumes. As RBVs encode trees with a fixed number of layers and sectors, they can be directly used as a representation to train neural networks. Specifically, we are interested in learning how to map the semantic mask of a tree and its RBV. A network for solving this task can thus be defined as:

$$f_{RBV}(M) : \mathcal{M} \rightarrow \mathcal{R}, \tag{1}$$

where $M \in \mathcal{M}$ denotes the semantic mask and $\mathcal{R}$ the RBVs. Our goal is to estimate the values of the RBVs by using a neural network trained to solve regression. As the regression of a larger number of parameters is usually error-prone, we jointly train up to five output branches (heads) to predict the values of RBVs of different resolutions (Fig. 3). The regression of values for a low-resolution RBV helps supervise the regression of the values for a higher-resolution RBV. Therefore, we define RBVs with up to five levels of resolution.

For the first level, we define an RBV as one layer with one sector (i.e., a cylinder). We then progressively split this volume to generate RBVs with: 2 layers and 2 sectors (level 2, 4 values), 4 layers and 4 sectors (level 3, 16 values), 8 layers with 8 sectors (level 4, 64 values), and 16 layers with 16 sectors (level 5, 256 values). To train the network as a cascade of outputs, we concatenate the output of the head that predicts a coarser resolution with the image embedding and use it as the input for the head with higher resolution. Once trained, we can then jointly obtain the values for each of the RBV resolutions.

The neural network f_{RBV} that is trained to estimate RBVs as a cascade of four heads is shown in Fig. 6a. The input to our network is semantic masks that we split into three layers to represent each of the classes' background, branches, and foliage. The output sector values of the RBV that we use as labels to train our RBV CNN are normalized per each species – a scaling factor to obtain the final tree size is stored in the species parameter set P. We embed the masks with a lightweight ConvNet architecture (similar to AlexNet [Krizhevsky et al. 2012]) to extract image features (Fig. 6b).

Specifically, we use 7 layers of convolution and down-sampling operations to obtain a 512-wide image embedding vector. We then add three fully connected layers to define our final embedding vector and use it as input for each of the four decoder heads. Each head is trained to reconstruct the RBV values of a specific resolution.

4.2.4 Species Identification. It is also important to obtain the species of a tree for the fully automatic reconstruction of the tree models from photographs. We aim to learn the mapping from the semantic masks of a tree to a species identifier U. This neural network can be defined as:

$$f_{Species}(M) : \mathcal{M} \rightarrow \mathcal{U},$$

where $M \in \mathcal{M}$ denotes the semantic mask, and $U \in \mathcal{U}$ denotes the species identifier. We again implement this network as a ConvNet architecture that extracts image features from the semantic masks (Fig. 6b). On top of the image embedding, we train three fully connected layers with an output layer for classification (Fig. 6c). Once the species identifier has been obtained, we use it to select a set of predefined parameters P that can be used with our developmental model to grow trees.

4.3 Bi-Modal Tree Development

Our developmental model provides two modes of growth. First, we can express tree development using a phenomenological model of growth [Měch and Prusinkiewicz 1996; Stava et al. 2014]. Additionally, our method can express tree development as a process of self-organization of branches in space [Palubicki et al. 2009; Runions et al. 2007]. While the phenomenological growth mode simulates tree development according to observations from botany at the branch scale, the self-organizing mode is used to grow an overall realistic shape at the tree scale.

A tree model in our method is defined as an acyclic graph $G = \{N, E\}$ composed of nodes $n \in N$, edges $e \in E$ and a set of global parameters P_g. One node in G is the root node n_r. Each node n stores attributes defining the state of a branch segment of the tree model, such as the diameter, age, position in 3D space, and vigor. Tree development in our method is expressed by the addition, removal, and modification of attribute values of nodes during the simulation.

The **phenomenological growth** employs a growth function that assigns new states to nodes based on their current state in each simulation step. This function describes well-studied biological processes such as phyllotaxy, phototropism, gravitropism, branch shedding, hormonal control of buds, and branch reiteration. For a detailed description of the growth function we refer to [Měch and Prusinkiewicz 1996; Stava et al. 2014]; a list of the used parameters and their corresponding values for the tree species used in this paper are provided in Appendix (Tab. 3, 2).

In the case of **self-organizing growth**, we distribute markers A in a 3D bounding volume around the tree model. Growing branches are attracted towards markers in this space that are within a cone-shaped volume extended in the direction of growth. The markers within proximity of branches are removed. This results in branches avoiding growth into the same regions of space and facilitates branch competition for space. Branches can theoretically collide, but we did not observe any branch collisions for our tree models in practice.

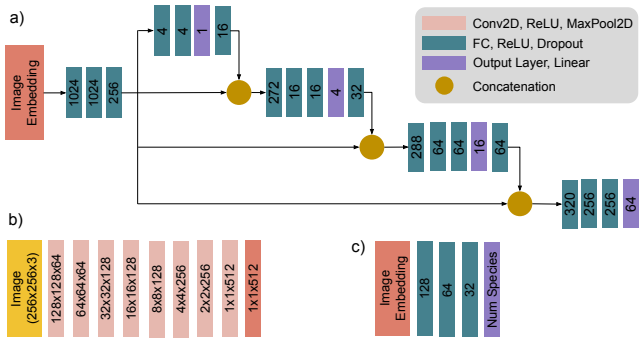

Fig. 6. Network architectures used in our framework. We use a cascade of dense layers to predict RBVs with four different resolutions on top of an image embedding (a). Input to this network is an image embedding feature vector that we obtain with a common ConvNet architecture (b). To identify the species of a tree we use the ConvNet architecture (b) to predict class probabilities with a few additional layers (c).

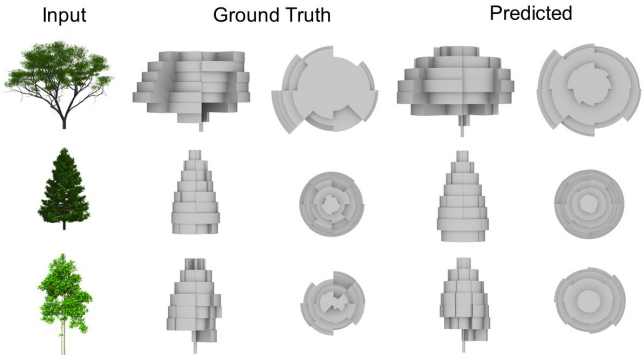

Fig. 7. Visualization of ground truth and predicted RBVs from the top and the side. Our RBV CNN allows us to closely reconstruct the 3D structure of a tree model from an input semantic segmentation mask.

Furthermore, we do not explicitly ensure that all space is occupied with branches. For more details on this algorithm, we refer to [Palubicki et al. 2009; Runions et al. 2007].

4.4 Reconstructing Trees with Bi-Modal Growth

The deep learning algorithms described in Sect. 4.2 return the semantic mask M of the tree that includes the main branches close to the trunk and the foliage. We then use the RBV CNN and the Species CNN to estimate the values of the RBV and a species identifier U from M. By default, the estimated RBV is normalized in vertical height. To account for species variation in height, we scale the RBV with an estimated species flag. The species identifier U is used to select a parameter set P of the corresponding species. We then use our novel bi-modal developmental model to grow a tree model with the parameters P. Finally, the RBV is used to spatially constrain the growth of the tree model so that it does not grow outside the RBV (Fig. 4.3).

Before generating tree geometry with the developmental model, we uniformly distribute markers A in the following way: first, we place a vertically oriented quad I_M, co-aligned with the x-axis, into a 3D scene and texture it with the mask M (Fig. 8a). Then, we place markers on the 2D quad I_M where the texture contains branch

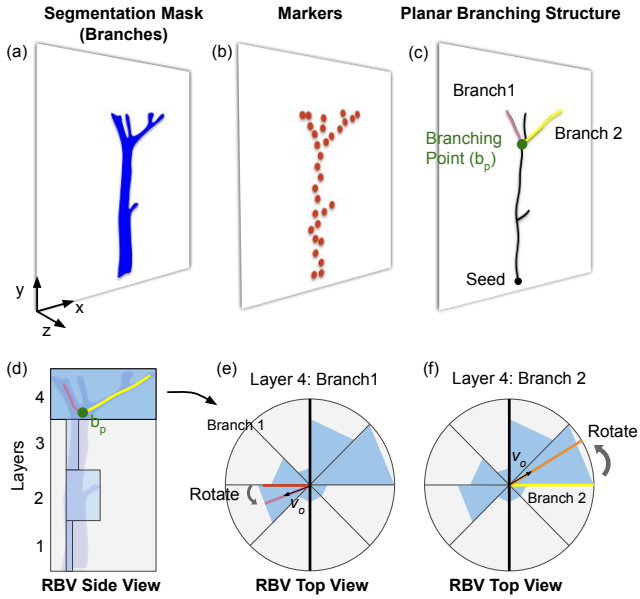

Fig. 8. The bi-modal growth model constraints tree growth with a segmentation mask M and a RBV. We use a segmentation mask of branches (a) to place markers in a 2D plane in 3D space by sampling the segmentation mask (b). We then use the markers to guide the growth process of a planar branching structure (c). To obtain a 3D branching structure, we rotate branches by determining a rotation vector based on the geometric center of sectors in the RBV (d, e, f). Depending on where a branching point is located in the RBV, we either use the left half (e) or the right half (f) of the RBV sectors to compute the geometric center and the resulting rotation vector v_o.

pixels (Fig. 8b). This results in generating a marker for each pixel. Positions of markers A are computed by placing them at positions that are covered by the corresponding blue pixels in M, which indicate branch occupancy. After this step, markers A are distributed on a 2D quad in 3D space according to the pixel distribution of M.

Once the 3D scene is filled with markers, we grow a model using the self-organizing mode of our developmental model. We initialize the growth direction with the up vector and place the seed at the position of the bottom-most marker A. Growth continues until all markers are consumed, and a planar branching structure emerges (Fig. 8c). Then, we reorient the planar branching structure into the 3D space of the RBV to obtain a natural branching structure (Fig. 8d). For each branching point $b_p \in G$, we determine the layer l_i containing it. In that layer level i, we select either the left or the right half of sectors to compute a geometric center g_c. We then compute a reorientation vector $v_o = b_p - g_c$ towards which we align the branch emerging from b_p by applying a rotation around the y axis (Fig. 8e, f). As the geometric center indicates the extent of sector volumes, branches are rotated towards the largest sectors for each layer. Note that the collisions are solved implicitly because branches will not grow into occupied space.

4.4.1 Tree Crown. Once the self-organizing growth mode is completed and the initial 3D branching structure has been generated, we switch to the phenomenological growth mode. We use the RBV as an

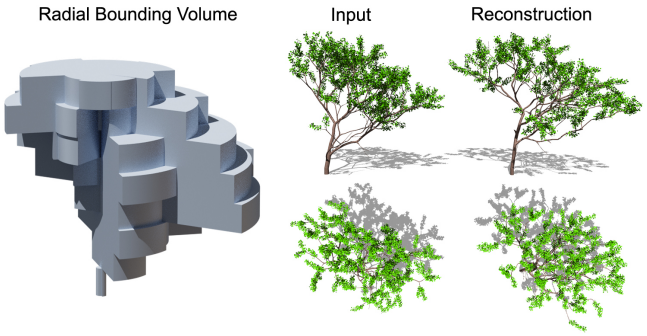

Fig. 9. RBVs can be used to represent and reconstruct intricate branching structures as required for asymmetric tree models.

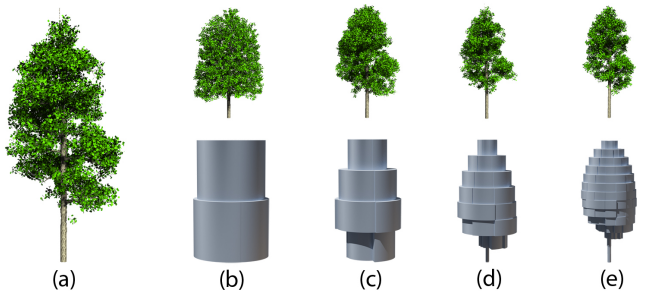

Fig. 10. Modeling results for different RBV levels. We use the input tree (a) and predict RBVs with the resolutions 2x2 (b), 4x4 (c), 8x8 (d), and 16x16 (e). A higher resolution allows us to more precisely constrain the growth and thereby to more faithfully reconstruct the input tree.

intersection volume – each outreaching branch is shed. This step is inspired by Open L-Systems [Měch and Prusinkiewicz 1996] to grow branches into fixed spatial envelopes. The development of a tree is completed once a predefined amount of nodes has been reached or the tree has been simulated to its maximum age. Eventually, the generated tree model is entirely inside the RBV, and its projection from the viewpoint corresponds to the segmentation mask M.

4.5 Dynamic Tree Models

To show the usefulness of our framework for content creation, we show how reconstructed trees can be readily combined with methods that enable modeling tree dynamics. We use Cosserat rods to efficiently simulate the dynamic behavior of the extracted tree graphs [Michels et al. 2015; Shao et al. 2021]. Each branch is discretized by arranging several nodes along the centerline, which are connected by rod segments. Similarly, adjacent branches are connected using rod segments. This requires storing additional attributes per node n such as velocity and angular velocity allowing for the convenient implementation of bending and twisting effects. Please refer to Pirk et al. [2017] for details utilizing position-based dynamics according to Kugelstadt and Schoemer [2016].

5 IMPLEMENTATION AND RESULTS

The developmental model and RBVs have been implemented in C++ with OpenGL. We have developed an interactive framework that generates a large number of trees for dataset generation. The neural networks described in Sect. 4.2 were implemented as separate frameworks in Python with Keras and Tensorflow backend. For

RBV Level	LxS	# Params.	Error GT	Error DL	Error RM	Error CO
1	1x1	1	1.6%	1.9%	1.7%	1.5%
2	2x2	4	1.7%	1.8%	1.9%	1.9%
3	4x4	16	3.1%	3.3%	3.4%	3.7%
4	8x8	64	3.5%	3.9%	4.0%	4.1%
5	16x16	256	4.2%	4.3%	4.4%	4.6%

Table 1. Error rates obtained with different network and training configurations: *Error GT* refers to the training of our cascaded network with the ground truth masks. The results of training the cascaded network with the predicted masks from DeepLab-V3 are shown as *Error DL*. *Error RM* denotes the results of training our network with concatenated semantic segmentation and RGB images. Finally, to validate the usefulness of our cascaded network, we performed an ablation study with a common CNN (8 conv and 3 dense layers) that is trained individually for each layer on the synthetic ground truth masks. The error rates for this experiment are shown as *Error CO*. LxS denotes the number of layers (L) and sectors (S).

DeepLab-V3 [Chen et al. 2016] we relied on the publicly available Python and Tensorflow implementation. We run our framework on an Intel i9-9900k at 4.8GHz with Nvidia GeForce RTX 2080.

5.1 Neural Network Training

We use the DeepLab-V3 network for semantic segmentation with the xception41 backbone and initialize it with weights pre-trained with the ImageNet data set. The atrous rates are set to (6, 12, 18), the output stride to 8, the learning rate to 0.0001, and the batch size to 2 (due to memory restrictions). The network was trained for 60K iterations on the 21K input images of our synthetic dataset, which took about 8 hours. The network achieves a mean IoU score of 74.7% across all classes and requires around 3.2 seconds for the interference of a single, high resolution ($1,280 \times 1,280$ pixels) segmentation mask.

The RBV CNN is trained with the Adam optimizer, a learning rate of 0.001, and a batch size of 32. We train the network against a Huber loss for regression [Hastie et al. 2001] on the segmentation mask M and the RBV R of the dataset described in Sect. 4.2.1. We split the RGB segmentation masks into three channels and normalize them into a $[0, 1]$ range. Similarly, we use normalized RBV sector values as labels to train our RBV CNN.

Tab. 1 summarizes the mean absolute error for different experiments: *Error GT* refers to the training of our cascaded network with the ground truth semantic segmentation masks. Training the cascaded network with the predicted masks from DeepLab-V3 is shown as *Error DL*. *Error RM* denotes the results for training a network on concatenated RGB images and semantic segmentation masks. Finally, to further validate the usefulness of our cascaded network, we performed an ablation study with a common CNN (8 convolutional and 3 dense layers) that was trained individually for each layer on the synthetic ground truth masks. The error rates for this experiment are shown as *Error CO*. While the relative error between *Error GT* and *Error DL* (0.4% at level 4) reveals the error introduced by the semantic segmentation network, the relative error between *Error GT* and *Error CO* (0.3% at level 4) shows that the cascaded network architecture is superior to the common training scheme.

We train this network for 4 hours, enabling us to infer RBV values at 17 ms. The species classification network was also trained with the Adam optimizer, a learning rate of 0.001 and a batch size of 32. We trained this network with a cross-entropy loss on our dataset with seven species as individual classes and obtained an accuracy of identifying a species of 94%. We trained this network for 1 hour and obtained species identifiers at 12 ms. The species identifier is then used to look up a predefined set of parameters P for each individual species. Each of the three networks is trained separately, but on the same dataset. Overall, it takes ten seconds on average to automatically reconstruct a 3D tree model with our framework.

Fig. 11. We reconstruct a tree model from a photograph and use Cosserat rods to simulate physically plausible sway motions. Here a user pulls and then releases a branch, which results in the typical sway motions of branching structures.

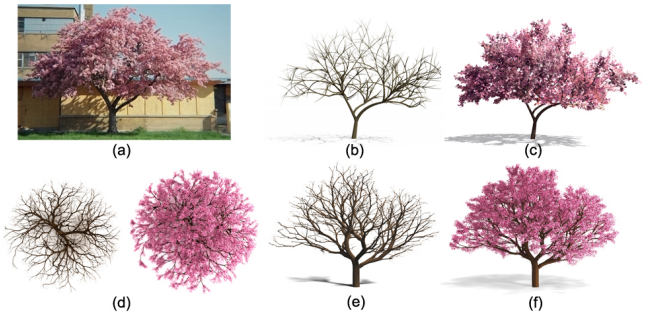

Fig. 12. Qualitative comparison to Tan et al. [2008]: given the input photograph provided in the previous work (a) along with reconstructed model (b, c), we reconstruct a similar tree model (d-f), with similar visual properties. In contrast to the previous work that requires users to sketch the segmentation mask, we automatically reconstruct the tree model in less than 3 seconds.

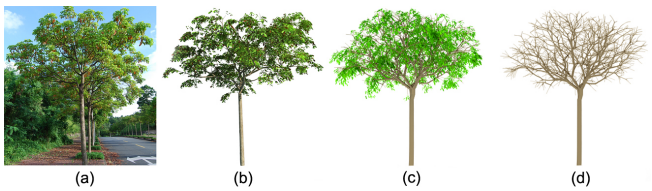

Fig. 13. Qualitative comparison to Livny et al. [2011]: while their method reconstructs a tree model (b) from 3D point clouds, we only use a single photograph (a) to reconstruct a tree model (c, d).

5.2 Results

Figs. 1, 14, and 15 show the reconstruction of real and synthetic trees with our pipeline. In both cases, we use the semantic segmentation network to detect and segment a tree in a single photograph. We then use the mask to identify the tree species, which – in turn – provides us with procedural model parameters. Additionally, we use the mask to estimate the trees' RBV. We then use the mask, RBV, and the procedural model parameters to grow branching structures similar to the tree shown in the input image. Depending on the used species, the number of branches of reconstructed models ranges from 2k to 10k.

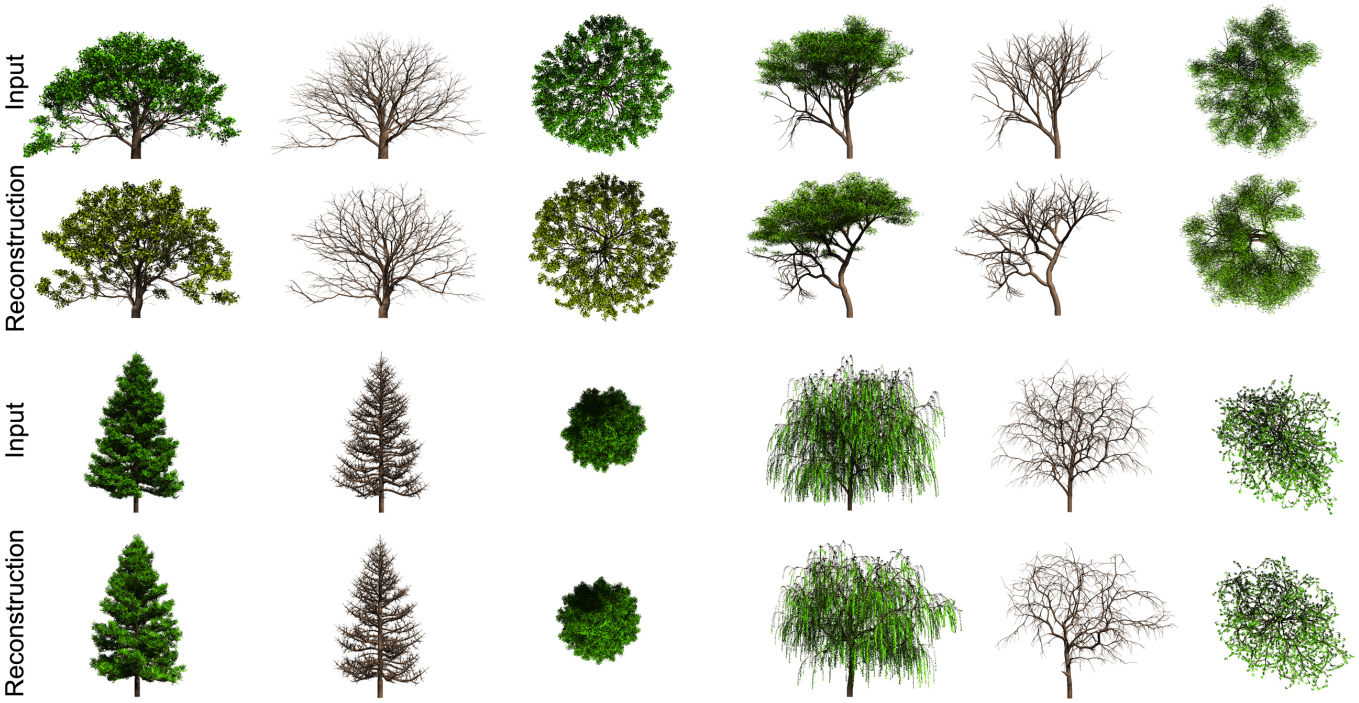

Fig. 14. Reconstructions of synthetically generated tree models of the four different species: Oak (upper left), Acacia (upper right), Pine (bottom, left), and Willow (bottom, right). Our method allows us to reconstruct the main branching structure and the overall shape with high visual fidelity. For each input tree model we show two reconstructed models generated with different random seeds and bi-modal growth with RBV and segmentation mask.

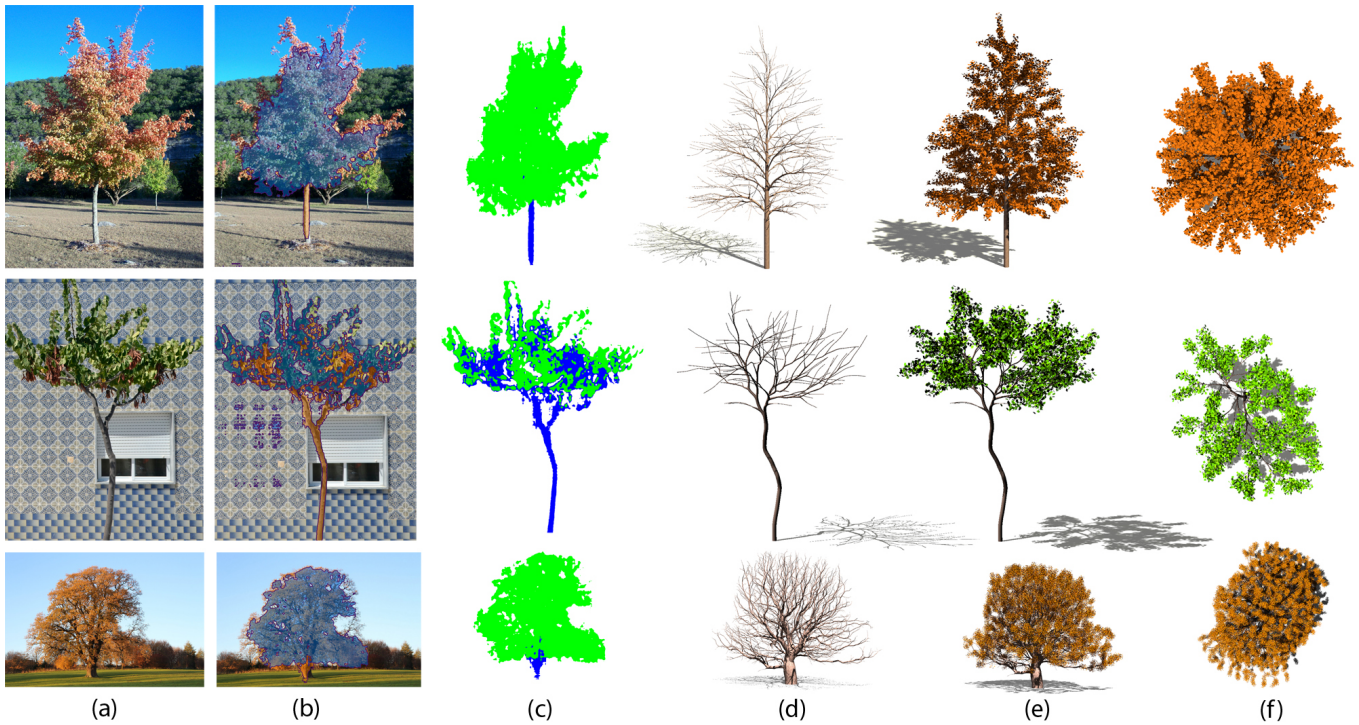

Fig. 15. Reconstruction fidelity for four different real trees. We use a neural network for semantic segmentation to obtain masks for branches and foliage from photographs (a)-(c). We then employ our novel radial bounding volume representation along with a dual-modal growth algorithm to reconstruct realistic branching structures (d). The resulting trees show visual features similar to what can be observed in the photographs (e, f).

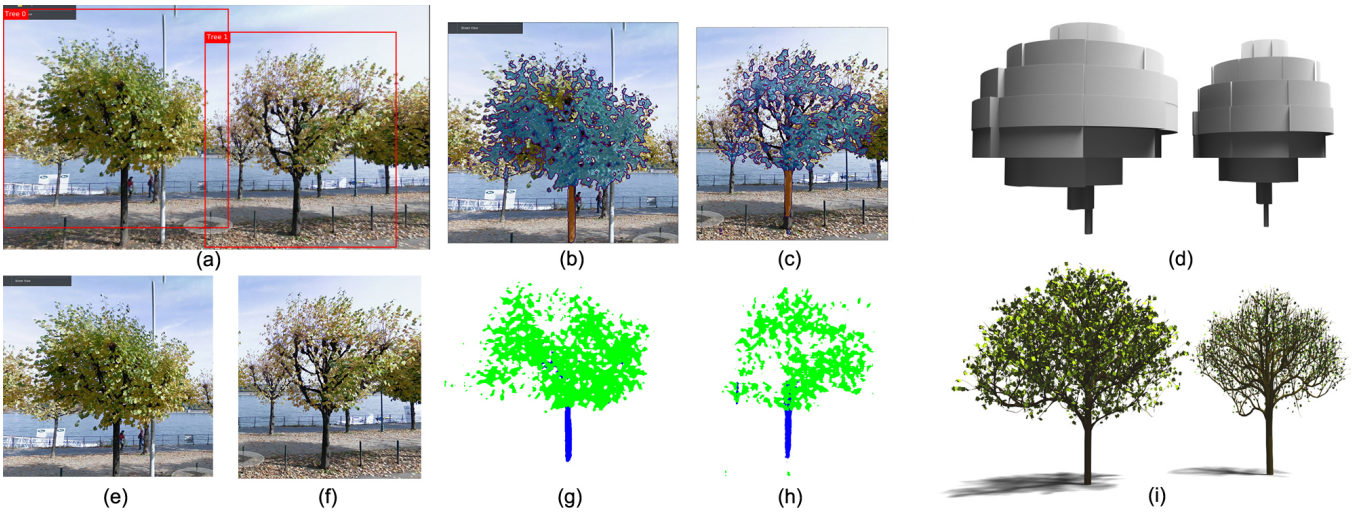

Fig. 16. Reconstructions of multiple trees: to reconstruct multiple trees we first detect bounding boxes of trees (a) to obtain cropped images (e, f). For each cropped image we can then compute the semantic segmentation masks (g, h) – overlayed with the rgb image in (b, c). We then predict RBVs (d) to reconstruct the tree models (i). Please note that we do not aim to learn the relative positioning of trees; the arrangement of the 3D scene is defined manually.

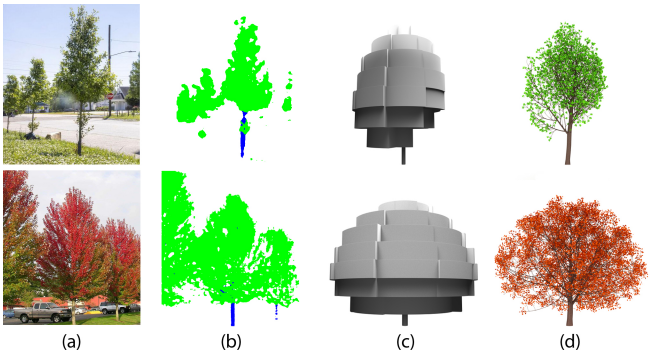

Fig. 17. Two failure cases: for the shown photographs (a), the semantic segmentation network was not able to fully separate foreground tree pixels from the background and other tree pixels (b). Consequently, the RBV network failed to predict the correct RBV (c), which in turn led to reconstructions that do not match the captured tree shape (d).

Fig. 10 shows how varying RBV levels affect tree growth. We take a synthetic tree with known geometry (Fig. 10a) and build several RBVs with varying resolutions (2×2, 4×4, and 8×8) and then we use our developmental model to grow the tree by using the RBVs.

Our method allows us to generate complex and detailed branching structures that can directly be used for animation. For the result shown in Fig. 11 we simulate rod dynamics to animate a tree model based on user interaction. A user pulls and then releases a branch, which causes the typical sway motions of trees. Based on our method, it is possible to reconstruct trees and immediately use them as animation-ready content in common production pipelines.

Figs. 1 and 9 show that RBVs are capable of representing complex asymmetric branching structures. For both tree models, the oak and the pine tree, the corresponding RBVs capture the main attributes of the branching structure, which is an essential property to reconstruct a tree model from a single image.

Figs. 12 and 13 show a qualitative comparison of our results to the state-of-the-art tree reconstruction approaches. For single image tree reconstruction, we show one of the results from Tan et al. [2008]. Given a single photograph of a Cherry tree, their method reconstructs a detailed tree model by providing user-defined sketches of the main branching structure and the crown shape. Unlike them, we use the photograph to reconstruct a tree model in only a few seconds automatically. Fig. 13 shows a comparison to the method of Livny et al. [2011] that relies on laser-scanned point sets – which are less convenient to obtain than single images – to reconstruct detailed tree models.

In Fig. 17 we show two failure cases. For the shown example photographs (a) the semantic segmentation network was not able to fully separate foreground tree pixels from the background and other tree pixels (b). Consequently, the RBV network failed to predict the correct RBV (c), which in turn led to reconstructions that do not match the captured tree shape (d).

Finally, in Fig. 16 we show an experiment for multi-tree reconstruction. As our method relies on identifying the semantic segmentation masks of trees to separate branch from leaf pixels, we first detect bounding boxes of trees (a) with an object detection network [Ren et al. 2015]. We then individually apply our pipeline to each of the generated cropped images (e, f) to first obtain the semantic segmentation masks (g, h), which are overlayed with the RGB image for reference in (b, c). The segmentation masks can then be used to generate the RBVs (d) and the reconstructed trees (i). Note that we do not aim at reconstructing scenes automatically, so the trees need to be positioned manually.

6 EVALUATION, DISCUSSION, AND LIMITATIONS

We evaluate our method by proposing a set of metrics designed to assess tree form similarity. For that purpose we use several geometric metrics employed in forestry to quantify trees [Blozan 2006; Watson

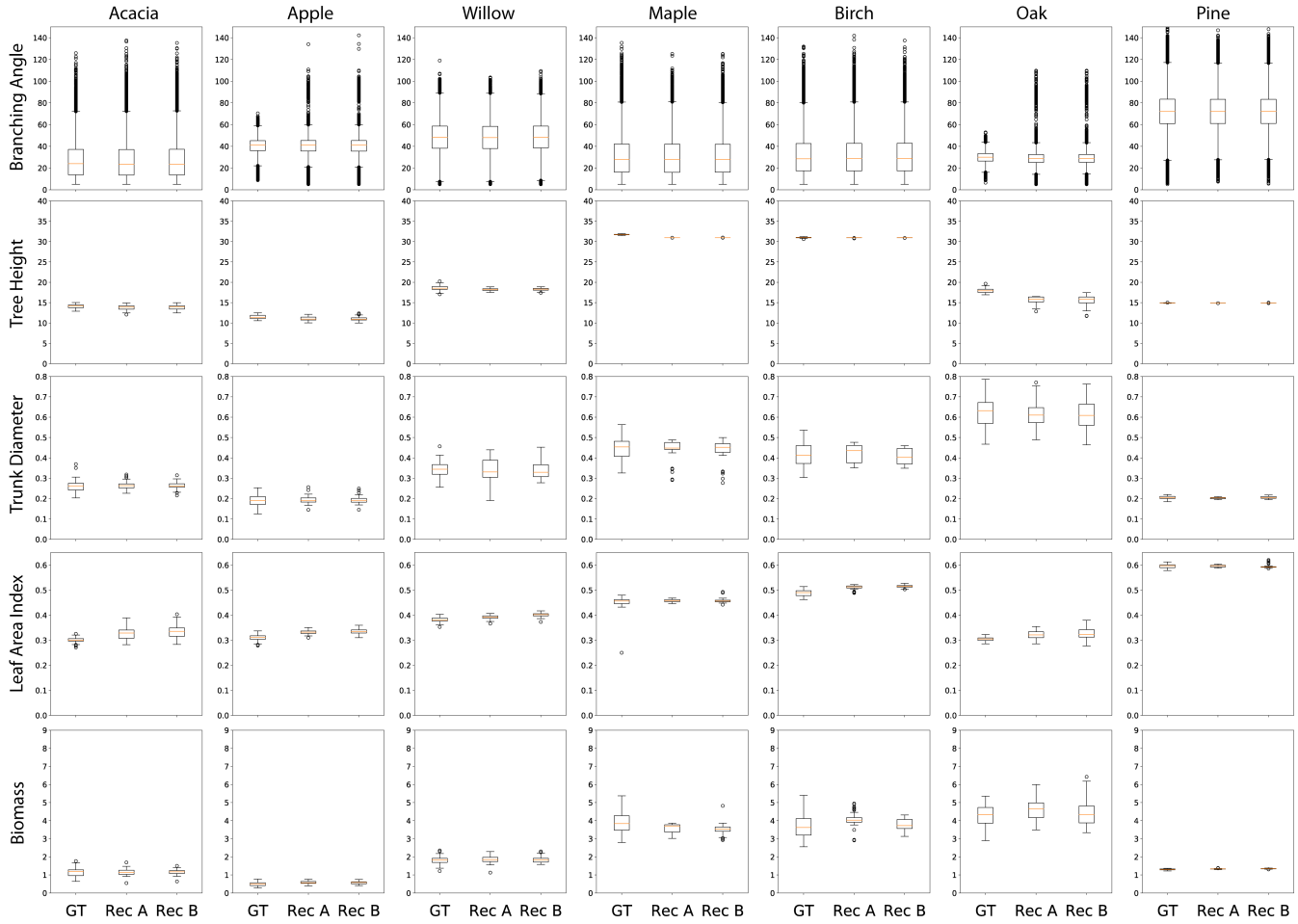

Fig. 18. Box plots of tree metrics: we assess 3D tree form similarity between ground truth data (GT), reconstructions from GT semantic segmentation masks and RBVs (Rec A), and reconstructions from predicted semantic segmentation masks and RBVs (Rec B). Rows indicate tree form metrics, such as branching angle, tree height, trunk diameter, LAI, and biomass. Columns indicate the virtual species used to create the synthetic data set, including acacia, apple, willow, maple, birch, oak, and pine. Overall, the correspondences of distribution means and variances are very close between GT and reconstructed data.

1947]. These metrics include average branching angle, trunk diameter (at base), tree height, total biomass, and leaf area index (LAI). We infer the total biomass by calculating the total volume of all branches. The LAI in our case is defined as the projected tree geometry on the ground per unit area.

We use the metrics to quantify the similarity between tree models. We use the ground truth tree models (GT) and compare them against the reconstructed tree models from the ground truth semantic segmentation masks and RBVs (Rec A). Additionally, we compute tree models from the predicted semantic segmentation masks and RBVs that we obtain from our neural networks (Rec B). In Fig. 20 we show boxplots of all the used metrics for all species in the synthetic dataset. Most of the boxplots indicate similar means and standard deviations between both GT and Rec A and GT and Rec B tree models. The overall correspondence between GT and Rec A results indicates that our bi-modal developmental model produces tree architectures similar to the phenomenological mode alone. However, not all reconstructions show a high degree of overlap. Most notably, maple and oak distributions of biomass show a small overlap resulting from the varying growth dynamics imposed by the self-organizing mode. The closely overlapping distributions between GT and Rec B results show the overall performance of the network inference on synthetic data.

As the metrics of branching angle, tree height, trunk diameter, leaf area index, and biomass represent global measures of tree shape, we additionally use a spatially local metric to assess tree geometry. We call it the maximum radial trunk distance (MRT). The MRT metric is calculated for a number of discrete vertical layers by finding the maximally distant branch node from the trunk in each layer. Fig. 20 shows the mean MRTs for all species and their standard error. These graphs give a visual impression of the overall variance and mean of tree geometry used in preparing our synthetic training dataset.

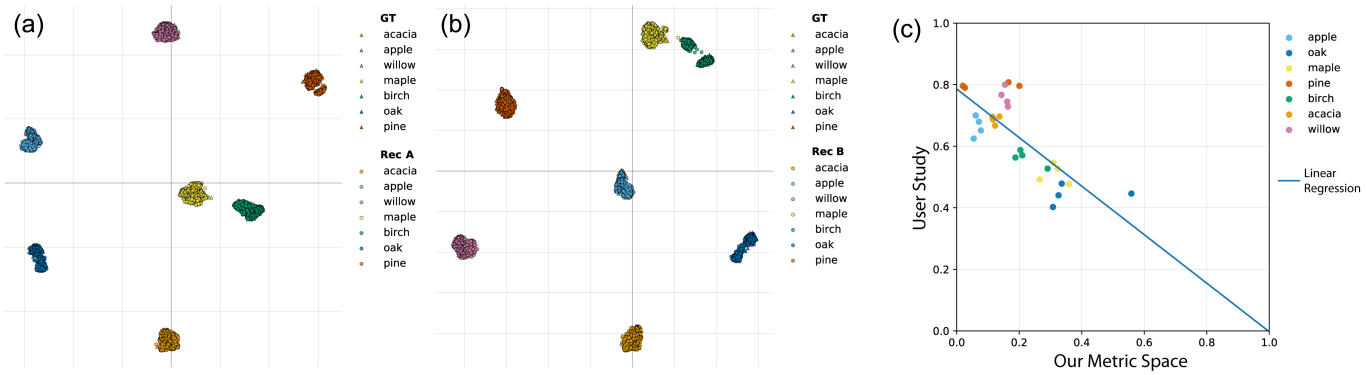

Fig. 19. t-SNE plots of the 10-dimensional metric space constructed from LAI, tree height, biomass, trunk diameter, branch angle and five vertical layers of MRTs. Ground truth tree models are shown as colored disks, Rec A (a), and Rec B (b) tree models as colored triangles. Overall, the distributions of GT, Rec A and Rec B tree models overlap for each species which indicates that our reconstruction algorithm allows for the faithful reconstruction of tree models. In (c) we show a graph of the user study ranking number (0-5) of GT to Rec B similarity and our normalized metric space distances. As shown by the linear regression line, the similarities are negatively correlated. This means, that tree models which are further apart in metric space received on average a lower ranking number in the user study and vice versa. Therefore, the correlation of the user study and the overlap of distributions in metric space shows that our pipeline generates visually similar reconstructions.

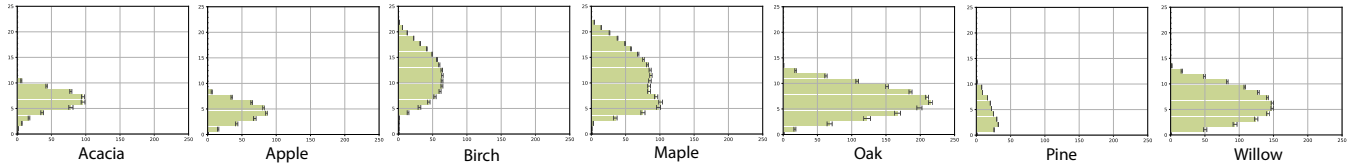

Fig. 20. MRTs of all species: we assess the variance and overall shape of tree models in the synthetic dataset used for NN training. Vertical axes indicate height of tree, horizontal axes the mean maximum distance of branch nodes to trunk. Vertical bars denote the standard deviation for a given species.

Furthermore, we use the global metrics (branching angle, tree height, trunk diameter, leaf area index, biomass) together with 5 vertical layers of MRT to construct a 10-dimensional metric space. Five axes in that metric space correspond to the five global metrics, while the remaining five axes are used to represent MRTs for five uniformly selected vertical layers. We then embed our training dataset and the estimated dataset into this metric space. The distances in metric space indicate the similarity of tree form between two tree models. In Fig. 19 we show the result of a non-linear dimensionality reduction using t-SNE to obtain 2D projections of our metric space.

6.1 User Study

We performed a user study to validate if the reconstructed trees are perceived as similar. We showed the source tree and the reconstructed trees to 100 users. We have asked the participants a question, "Do the trees look similar?" and they had to choose 0-strongly disagree, 1-disagree, 2-somehow disagree, 3-somehow agree, 4-agree, 5-strongly agree. We assigned a value of $0 - 0.2 - 0.4 - 0.6 - 0.8 - 1$ to each choice. We performed two tests: one with leaves and one of the same models without leaves. We also run separate trials for the reconstruction with the mask and without the mask. The tests were performed for each species using two images selected from the center of the cluster and two images that were far from the centroid (see Fig. 19). The hypothesis was that the trees close to the center would be evaluated as more similar than those far. We used Mechanical Turk, and we selected only certified Mechanical Turk Masters to make sure the answers are valid. *With leaves:* The users selected the trees as similar to the source in 63% with the slight preference for the trees far from the center 64% over the trees close to the center 61%. If the mask is used for the reconstruction, the perceived similarity is 63%. The reconstruction without a mask was at 61%. *Without leaves:* The users selected the trees as similar to the source in 59% with no preference about the tree distance from the center. There was no preference for the reconstruction with and without the mask (58.0 vs. 57.8).

In Fig. 19 (c) we show a graph of the user study ranking number (0-5) of GT to Rec B similarity and our normalized metric space distances. As shown by the linear regression line, the similarities are negatively correlated. This means, that tree models which are further apart in metric space received on average a lower ranking number in the user study and vice versa. Therefore, the correlation of the user study and the overlap of distributions in metric space shows that our pipeline generates visually similar reconstructions.

6.2 Discussion and Limitations

There exist three distinct methods to generate 3D tree models of real trees. First, trees can be reconstructed based on sensor data such as laser scanners [Livny et al. 2011] or multiple images [Neubert et al. 2007; Tan et al. 2007]. These methods are generally more accurate than our approach but rely on data that is costly to obtain

and not readily available compared to single photographs of trees. Second, various procedural or interactive approaches to tree growth have been proposed to generate plausible 3D tree architectures (e.g. xfrog, SpeedTree). However, the manual tuning of parameters is not intuitive and requires expert knowledge. Furthermore, manually tuning parameters to reconstruct trees shown in input photographs lacks control and is not feasible for applications that require the generation of a large collection of diverse 3D tree models. Finally, there exist semi-automatic single-image tree reconstruction methods such as proposed by Tan et al. [2008] that rely on user annotations. These methods are most closely related to our approach, but as these methods rely on manually generated masks, they do not support the reconstruction of 3D tree models at scale.

Our method is the first to explore the fully automatic reconstruction of trees from single photographs. While the quality of the tree models generated with our framework is of high visual fidelity. Our method also has several limitations. We have tested a wide range of data augmentation strategies that span from configuring rendering parameters, such as adjusting camera pose, shadow intensity, or a number of lights, to image transformations, such as brightness or color changes. The data augmentation helps to obtain masks for a large number of real trees of species used in our paper.

Robustly bridging the domain gap by training neural networks on synthetic data to work on real photographs of arbitrary species remains a challenging problem. Furthermore, we used semantic segmentation masks – instead of RGB images – to train the RBV and Species CNNs to avoid overfitting the networks on synthetic data. The semantic masks encode the prominent features of tree models, which is sufficient to obtain species identifier and RBV values. Thus, semantic masks serve as a representation to help overcome the domain gap between real and synthetic data.

Another limitation of our method is that we cannot meaningfully encode trees with disjoint horizontal branching structures. RBVs only store a single distance value per sector, which limits encoding more complex tree shapes. This representation could be extended by storing multiple values (e.g., for ranges) or by discretizing the spatial extend into a number of buckets. However, extending the representation would result in increasing the number of values that need to be learned and therefore – potentially – reduce the reconstruction accuracy.

7 CONCLUSION AND FUTURE WORK

We have introduced a novel method for reconstructing trees from single photographs. This enables the large-scale reconstruction of tree models which is essential for many applications in urban reconstruction or games and movies. In contrast to previous work, our approach is fully automatic and does not require any user intervention. This advances the state-of-the-art of tree modeling at scale in computer graphics. Instead of employing user-defined sketches, we use a pipeline of neural networks to obtain semantic masks of trees, to identify their species, and to estimate their 3D structure from 2D photographs – a challenging and ill-posed problem. We have introduced *radial bounding volumes* as a lightweight and fixed-size representation for tree models that facilitates learning tree form. Growing trees into an RBV while constraining their development with the obtained semantic masks enables us to carefully guide tree growth so as capture the defining details of real trees. We have extensively evaluated our method based on a number of quantitative metrics to assess tree form. Our experiments indicate that we can successfully reconstruct complex 3D branching structures across different tree instances and across a variety of tree species. Finally, we have shown that the generated branching structures can directly be animated with existing methods for rod dynamics. This way we provide an end-to-end method for content creators that allows us to create animation-ready plants from photographs.

Given the current state of our framework multiple avenues of future work seem possible. For one, it seems interesting to relax the constraint to only reconstruct trees from a single image. While we have shown that this is successful for a number of different trees shapes, it seems plausible that a collection of images, e.g. as obtained from video sequences, can help to further improve the reconstruction. Moreover, instead of just focusing on trees, it seems promising to extend our reconstruction pipeline to model a larger variety of plants, including individual organs or plant parts. Finally, it seems interesting to explore neural network architectures along with procedural models to reconstruct collections of plants.

ACKNOWLEDGMENTS

This research was funded in part by National Science Foundation grant #10001387, *Functional Proceduralization of 3D Geometric Models*. This research was supported by the Foundation for Food and Agriculture Research Grant ID: 602757 to Benes. The content of this publication is solely the responsibility of the authors and does not necessarily represent the official views of the foundation for Food and Agriculture Research. Klein and Michels gratefully acknowledge the baseline funding from of the Computational Sciences Group within KAUST's Visual Computing Center.

REFERENCES

F. Anastacio, M. C. Sousa, F. Samavati, and J. A. Jorge. 2006. Modeling Plant Structures Using Concept Sketches. In *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering (NPAR '06)*. Association for Computing Machinery, 105–113.

M. Aono and T.L. Kunii. 1984. Botanical Tree Image Generation. *IEEE Comput. Graph. Appl.* 4(5) (1984), 10–34.

O. Argudo, A. Chica, and C. Andujar. 2016. Single-picture Reconstruction and Rendering of Trees for Plausible Vegetation Synthesis. *Comput. Graph.* 57, C (2016), 55–67.

S. Behrendt, C. Colditz, O. Franzke, J. Kopf, and O. Deussen. 2005. Realistic real-time rendering of landscapes using billboard clouds. *Comp. Graph. Forum* 24, 3 (2005), 507–516.

B. Benes and E. U. Millán. 2002. Virtual Climbing Plants Competing for Space. In *Proceedings of the Computer Animation (CA '02)*. IEEE Computer Society, USA, 33.

W. Blozan. 2006. Tree Measuring Guidelines of the Eastern Native Tree Society. (2006).

D. Bradley, D. Nowrouzezahrai, and P. Beardsley. 2013. Image-based Reconstruction and Synthesis of Dense Foliage. *ACM Trans. on Grap.* 32, 4, Article 74 (2013), 74:1–74:10 pages.

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. 2016. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *CoRR* abs/1606.00915 (2016).

X. Chen, B. Neubert, Y.-Q. Xu, O. Deussen, and S. B. Kang. 2008. Sketch-Based Tree Modeling Using Markov Random Field. *ACM Trans. on Grap.* 27, 5, Article 109 (Dec. 2008), 9 pages.

O. Deussen, C. Colditz, M. Stamminger, and G. Drettakis. 2002. Interactive Visualization of Complex Plant Ecosystems. *VIS '02* (2002), 219–226.

N. Greene. 1989. Voxel Space Automata: Modeling with Stochastic Growth Processes in Voxel Space. *SIGGRAPH Comp. Graph.* 23, 3 (1989), 175–184.

J. Guo, H. Jiang, B. Benes, O. Deussen, X. Zhang, D. Lischinski, and H. Huang. 2020. Inverse Procedural Modeling of Branching Structures by Inferring L-Systems. *ACM Trans. on Grap.* 39, 5, Article 155 (June 2020), 13 pages.

R. Habel, A. Kusternig, and M. Wimmer. 2009. Physically Guided Animation of Trees. *Comp. Graph. Forum* 28, 2 (2009), 523–532.
T. Hädrich, B. Benes, O. Deussen, and S. Pirk. 2017. Interactive Modeling and Authoring of Climbing Plants. *Comput. Graph. Forum* 36, 2 (2017), 49–61.
T. Hastie, R. Tibshirani, and J. Friedman. 2001. *The Elements of Statistical Learning*. Springer New York Inc., New York, NY, USA.
H. Honda. 1971. Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body. *Journal of Theoretical Biology* 31, 2 (1971), 331 – 338.
T. Ijiri, S. Owada, and T. Igarashi. 2006. Seamless Integration of Initial Sketching and Subsequent Detail Editing in Flower Modeling. *Comp. Graph. Forum* 25, 3 (2006), 617–624.
B. Karis. 2013. *Real Shading in Unreal Engine 4.* Technical Report. Epic Games.
Y. Kawaguchi. 1982. A Morphological Study of the Form of Nature. *SIGGRAPH Comp. Graph.* 16, 3 (1982), 223–232.
J. Kratt, Mark Spicker, A. Guayaquil, M. Fišer, S. Pirk, O. Deussen, J. C. Hart, and B. Benes. 2015. Woodification: User-Controlled Cambial Growth Modeling. *CGF* 34, 2 (2015), 361–372.
A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, Vol. 25. Curran Associates, Inc., 1097–1105.
T. Kugelstadt and E. Schoemer. 2016. Position and Orientation Based Cosserat Rods. In *Proceedings of the 2016 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association.
C. Li, O. Deussen, Y.-Z. Song, P. Willis, and P. Hall. 2011. Modeling and Generating Moving Trees from Video. *ACM Trans. on Grap.* 30, 6, Article 127 (2011), 127:1–127:12 pages.
Y. Li, X. Fan, N. J. Mitra, D. Chamovitz, D. Cohen-Or, and B. Chen. 2013. Analyzing Growing Plants from 4D Point Cloud Data. *ACM Trans. on Grap.* 32, 6, Article 157 (2013), 10 pages.
B. Lintermann and O. Deussen. 1999. Interactive Modeling of Plants. *IEEE Comput. Graph. Appl.* 19, 1 (1999), 56–65.
Y. Livny, S. Pirk, Z. Cheng, F. Yan, O. Deussen, D. Cohen-Or, and B. Chen. 2011. Texture-lobes for Tree Modelling. *ACM Trans. on Grap.* 30, 4, Article 53 (2011), 10 pages.
S. Longay, A. Runions, F. Boudon, and P. Prusinkiewicz. 2012. TreeSketch: interactive procedural modeling of trees on a tablet. In *Proc. of the Intl. Symp. on SBIM*. 107–120.
D. L. Michels, J. P. T. Mueller, and G. A. Sobottka. 2015. A physically based approach to the accurate simulation of stiff fibers and stiff fiber meshes. *Comput. Graph.* 53 (2015), 136–146.
R. Měch and P. Prusinkiewicz. 1996. Visual models of plants interacting with their environment. In *Proc. of SIGGRAPH*. ACM, 397–410.
B. Neubert, T. Franken, and O. Deussen. 2007. Approximate Image-based Tree-modeling Using Particle Flows. *ACM Trans. on Grap.* 26, 3, Article 88 (2007).
B. Neubert, S. Pirk, O. Deussen, and C. Dachsbacher. 2011. Improved Model- and View-Dependent Pruning of Large Botanical Scenes. *Comp. Graph. Forum* 30, 6 (2011), 1708–1718.
M. Okabe, S. Owada, and T. Igarashi. 2007. Interactive Design of Botanical Trees Using Freehand Sketches and Example-based Editing. In *ACM SIGGRAPH Courses*. ACM, Article 26.
P. E. Oppenheimer. 1986. Real time design and animation of fractal plants and trees. *Proc. of SIGGRAPH* 20, 4 (1986), 55–64.
W. Palubicki, K. Horel, S. Longay, A. Runions, B. Lane, R. Měch, and P. Prusinkiewicz. 2009. Self-organizing Tree Models for Image Synthesis. *ACM Trans. on Grap.* 28, 3, Article 58 (2009), 10 pages.
S. Pirk, B. Benes, T. Ijiri, Y. Li, O. Deussen, B. Chen, and R. Měch. 2016. Modeling Plant Life in Computer Graphics. In *ACM SIGGRAPH 2016 Courses*. Article 18, 180 pages.
S. Pirk, M. Jarząbek, T. Hädrich, D. L. Michels, and W. Palubicki. 2017. Interactive Wood Combustion for Botanical Tree Models. *ACM Trans. on Grap.* 36, 6, Article 197 (Nov. 2017), 12 pages.
S. Pirk, T. Niese, O. Deussen, and B. Neubert. 2012a. Capturing and animating the morphogenesis of polygonal tree models. *ACM Trans. on Grap.* 31, 6, Article 169 (2012), 10 pages.
S. Pirk, T. Niese, T. Hädrich, B. Benes, and O. Deussen. 2014. Windy Trees: Computing Stress Response for Developmental Tree Models. *ACM Trans. on Grap.* 33, 6, Article 204 (2014), 11 pages.
S. Pirk, O. Stava, J. Kratt, M. A. M. Said, B. Neubert, R. Měch, B. Benes, and O. Deussen. 2012b. Plastic trees: interactive self-adapting botanical tree models. *ACM Trans. on Grap.* 31, 4, Article 50 (2012), 10 pages.
P. Prusinkiewicz and Aristid Lindenmayer. 1990. *The Algorithmic Beauty of Plants*. Springer-Verlag New York, Inc.
L. Quan, P. Tan, G. Zeng, L. Yuan, J. Wang, and S. B. Kang. 2006. Image-Based Plant Modeling. *ACM Trans. on Grap.* 25, 3 (July 2006), 599–604.
E. Quigley, Y. Yu, J. Huang, W. Lin, and R. Fedkiw. 2018. Real-Time Interactive Tree Animation. *IEEE Trans. on Vis. and Comp. Graphics* 24, 5 (2018), 1717–1727.
A. Reche-Martinez, I. Martin, and G. Drettakis. 2004. Volumetric reconstruction and interactive rendering of trees from photographs. *ACM Trans. on Grap.* 23, 3 (2004), 720–727.
W. T. Reeves and R. Blau. 1985. Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems. *SIGGRAPH Comput. Graph.* 19, 3 (July 1985), 313–322.
S. Ren, K. He, R. Girshick, and J. Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc.
A. Runions, B. Lane, and P. Prusinkiewicz. 2007. Modeling Trees with a Space Colonization Algorithm. *EG Nat. Phenom.* (2007), 63–70.
H. Shao, T. Kugelstadt, T. Hädrich, W. Pałubicki, J. Bender, S. Pirk, and D. L. Michels. 2021. Accurately Solving Physical Systems with Graph Learning.
A. R. Smith. 1984. Plants, fractals, and formal languages. In *Proc. of SIGGRAPH*. ACM Press, 1–10.
O. Stava, S. Pirk, J. Kratt, B. Chen, R. Měch, O. Deussen, and B. Benes. 2014. Inverse Procedural Modelling of Trees. *Comp. Graph. Forum* 33, 6 (2014), 118–131.
P. Tan, T. Fang, J. Xiao, P. Zhao, and L. Quan. 2008. Single Image Tree Modeling. *ACM Trans. on Grap.* 27, 5, Article 108 (2008), 7 pages.
P. Tan, G. Zeng, J. Wang, S. B. Kang, and L. Quan. 2007. Image-based Tree Modeling. *ACM Trans. on Grap.* 26, 3, Article 87 (2007).
B. Wang, Y. Zhao, and J. Barbič. 2017b. Botanical Materials Based on Biomechanics. *ACM Trans. on Grap.* 36, 4, Article 135 (July 2017), 13 pages.
G. Wang, H. Laga, J. Jia, N. Xie, and H. Tabia. 2018a. Statistical Modeling of the 3D Geometry and Topology of Botanical Trees. *Comp. Graph. Forum* 37, 5 (2018), 185–198.
G. Wang, H. Laga, N. Xie, J. Jia, and H. Tabia. 2018b. The Shape Space of 3D Botanical Tree Models. *ACM Trans. on Grap.* 37, 1, Article 7 (Jan. 2018), 18 pages.
Y. Wang, X. Xue, X. Jin, and Z. Deng. 2017a. Creative Virtual Tree Modeling Through Hierarchical Topology-Preserving Blending. *IEEE Trans. on Vis. and Comp. Graphics* 23, 12 (2017), 2521–2534.
D. J. Watson. 1947. Comparative Physiological Studies on the Growth of Field Crops: I. Variation in Net Assimilation Rate and Leaf Area between Species and Varieties, and within and between Years. *Annals of Botany* 11, 1 (01 1947), 41–76.
J. Wither, F. Boudon, M.-P. Cani, and C. Godin. 2009. Structure from silhouettes: a new paradigm for fast sketch-based design of trees. *Comp. Graph. Forum* 28, 2 (2009), 541–550.
H. Xu, N. Gossett, and B. Chen. 2007. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Trans. on Grap.* 26, 4 (2007), Article 19, 13 pages.
Y. Zhao and J. Barbič. 2013. Interactive Authoring of Simulation-ready Plants. *ACM Trans. on Grap.* 32, 4, Article 84 (2013), 12 pages.

APPENDIX

We use the phenomenological growth model proposed by Stava et al. [2014] to simulate tree growth. The parameters used for this model are shown in Tab. 3. In Tab. 2 we provide the parameter values used to generate tree models in our dataset. Once these values are defined, tree models can be generated automatically. In Fig. 21 we show example tree models for each species. Leaf and bark textures as well as the size of leaves were selected manually for each species. We show additional results of reconstructed tree models from single photographs in Fig. 22.

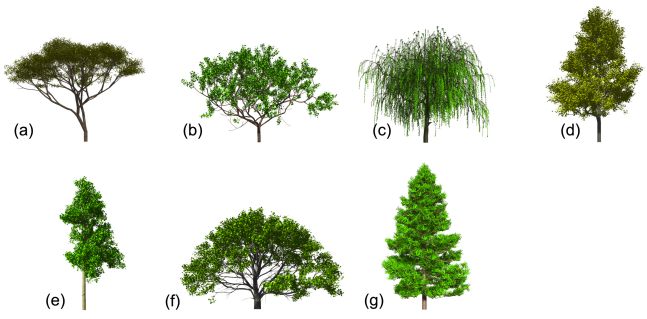

Fig. 21. Examples tree models of seven different species: (a) Acacia, (b) Apple, (c) Willow, (d) Maple, (e) Birch, (f) Oak, (g) Pine.

Params.	Acacia	Apple	Willow	Maple	Birch	Oak	Pine
G_{NLB}	2	2	2	2	2	3	3
G_{AAV}	2	20	12	2	5	20	0
G_{BAM}	20	45	43	30	30	29	80
G_{BAV}	5	2	3	5	5	2	4
G_{RAM}	113	91	80	130	130	91	10
G_{RAV}	13	1	4	30	30	1	30
F_{AKP}	0.9	0	0	0	0	0	0
F_{LKP}	0.006	0.21	0.21	0.01	0.01	0.21	0.015
F_{ALF}	1	0.39	0.54	0.85	0.85	0.39	0.04
F_{LLF}	1	1.13	0.94	0.2	0.2	1	0.1
F_{ADB}	3.5	3.13	0.38	4.87	4.87	3.13	0.01
F_{ADF}	0.9	0.13	0.9	0.98	0.98	0.13	0.9
F_{AAF}	0.9	0.82	0.31	0.42	0.42	0.82	0.87
F_{GR}	1.5	2.98	2.3	4.25	4.25	3	3.26
F_{ILB}	0.92	0.55	0.8	0.93	0.93	1	0.4
F_{IAF}	0.96	0.97	0.98	0.95	0.93	0.93	0.96
F_{ACB}	0.94	2.2	3.25	3.64	3.64	2.2	6.2
F_{AAF}	0.92	0.5	0.7	0.87	0.89	0.5	0.9
F_{ALF}	1	1	1	0.92	0.93	1	1
F_{MBA}	10	20	8	8	8	20	10
E_{LBF}	0.5	0.37	0.4	0.6	0.6	0.5	0.3
E_{PHO}	0.42	0.5	0.15	0.45	0.45	0.32	0.4
E_{GGB}	0.6	-0.54	-0.16	0.2	-0.33	0.08	0.05
E_{GLF}	-0.2	0.17	-0.09	0	0.35	-0.03	0.01
E_{PPF}	0.05	0.7	0.8	0.1	0.1	0.7	0.12
E_{LPF}	0.64	1.3	2.9	4.5	6	1.3	1
E_{GBS}	0.2	0.68	0.14	0.13	0.85	0.72	0.94
E_{GAF}	0.85	0.8	0.85	0.93	0.91	0.83	0.85
t	15	11	14	10	11	10	15

Table 2. Parameter values for each species used in our framework.

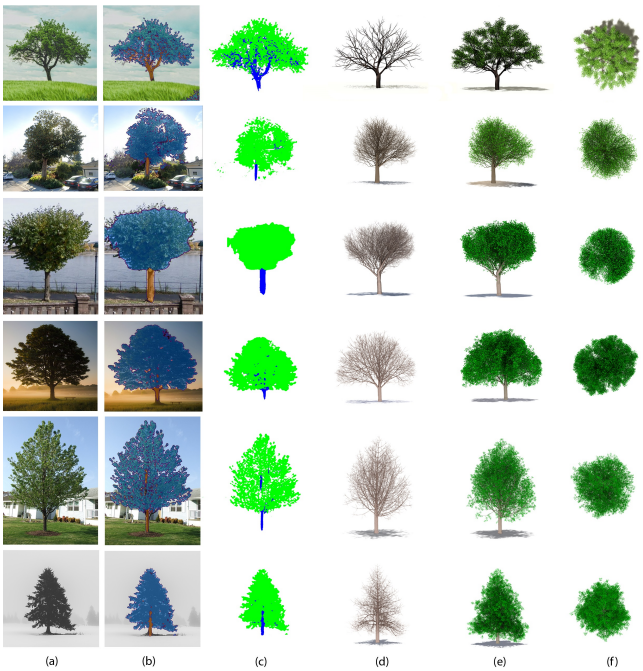

Fig. 22. Additional results of reconstructed models from single photographs: (a) input photograph, (b) semantic segmentation mask overlay, (c) semantic segmentation mask, (d) generated branching structure, (e) generated tree model (f) generated tree model top view.

Params.	Name	Description
G_{NLB}	Number of lateral buds	The number of lateral buds created per internode during growth.
G_{AAV}	Apical angle variance	The variance of the angle between the direction of parent shoot and the direction of apical bud.
G_{BAM}	Branching angle mean	The mean of angle between the direction of parent shoot and the direction of lateral bud.
G_{BAV}	Branching angle variance	The variance of angle between the direction of parent shoot and the direction of lateral bud.
G_{RAM}	Roll angle mean	The mean of orientation angle between two lateral buds created with the same internode.
G_{RAV}	Roll angle variance	The variance of orientation angle between two lateral buds created with the same internode.
F_{AKP}	Apical bud kill probability	The probability that a given apical bud will die during a growth cycle.
F_{LKP}	Lateral bud kill probability	The probability that a given lateral bud will die during a growth cycle.
F_{ALF}	Apical bud lighting factor	The influence of the lighting condition on the growth probility of a apical bud.
F_{LLF}	Lateral bud lighting factor	The influence of the lighting condition on the growth probility of a lateral bud.
F_{ADB}	Apical dominance base	The base level of auxin produced to inhibit parent shoots from growing.
F_{ADF}	Apical dominance distance factor	The reduction of auxin due to the transimission along parent shoots.
F_{AAF}	Apical dominance age factor	The reduction of auxin due to increasing age of the tree.
F_{GR}	Growth rate	The expected number of internodes generated along the branch during a growth cycle.
F_{ILB}	Internode length base	The base distance between two adjacent internodes on the same shoot.
F_{IAF}	Internode length age factor	The relation between internode length and age of the tree.
F_{ACB}	Apical control base	The impact of the branch level on the growth rate.
F_{AAF}	Apical control age factor	The relation of the apical control to the tree age.
F_{ALF}	Apical control level factor	The relation of the apical control to the branch level.
F_{MBA}	Max bud age	The maximum life time of a bud before forming a new shoot.
E_{LBF}	Light blocking factor	How much light will be blocked by branches and leaves.
E_{PHO}	Phototropism	The impact of the average growth direction of incoming light.
E_{GGB}	Gravitropism base	The impact of the average growth direction of the gravity.
E_{GLF}	Gravitropism level factor	The relation of the gravitropism and branch level.
E_{PPF}	Pruning factor	The impact of the amount of incoming light on the shedding of branches.
E_{LPF}	Low branch pruning factor	The height below which all lateral branches are pruned.
E_{GBS}	Gravity bending strength	The impact of gravity on branch structural bending.
E_{GAF}	Gravity bending angle factor	The relation of gravity bending related to the thickness of the branch.
t	Desired age	The expected age of the tree.

Table 3. Table of parameters for growth model