Realistic Modeling of Tree Ramifications from an Optimal Manifold Control Mesh

Zhengyu Huang, Zhiyi Zhang, Nan Geng, Long Yang, Dongjian He, and Shaojun Hu *

College of Information Engineering, Northwest A&F University, Yangling, China
{hzy823@163.com, hsj@nwsuaf.edu.cn}

Abstract. Modeling realistic branches and ramifications of trees is a challenging task because of their complex geometric structures. Many approaches have been proposed to generate plausible tree models from images, sketches, point clouds, and botanical rules. However, most approaches focus on a global impression of trees, such as the topological structure of branches and arrangement of leaves, without taking continuity of branch ramifications into consideration. To model a complete tree quadrilateral mesh (quad-mesh) with smooth ramifications, we propose an optimization method to calculate a suitable control mesh for Catmull–Clark subdivision. Given a tree's skeleton information, we build a local coordinate system for each joint node, and orient each node appropriately based on the angle between a parent branch and its child branch. Then, we create the corresponding basic ramification units using a cuboid-like quad-mesh, which is mapped back to the world coordinate. To obtain a suitable manifold initial control mesh as a main mesh, the ramifications are classified into main and additional ramifications, and a bottom-up optimization approach is applied to adjust the positions of the main ramification units when they connect their neibour. Next, the first round of Catmull–Clark subdivision is applied to the main ramifications. The additional ramifications, which were selected to alleviate visual distortion in the preceding step, are added back to the main mesh using a cut-paste operation. Finally, the second round of Catmull–Clark subdivision is used to generate the final quad-mesh of the entire tree. The results demonstrated that our method generated a realistic tree quad-mesh effectively from different tree skeletons.

Keywords: Tree quad-mesh, Construction optimization, Catmull–Clark subdivision, Manifold tree modeling

1 Introduction

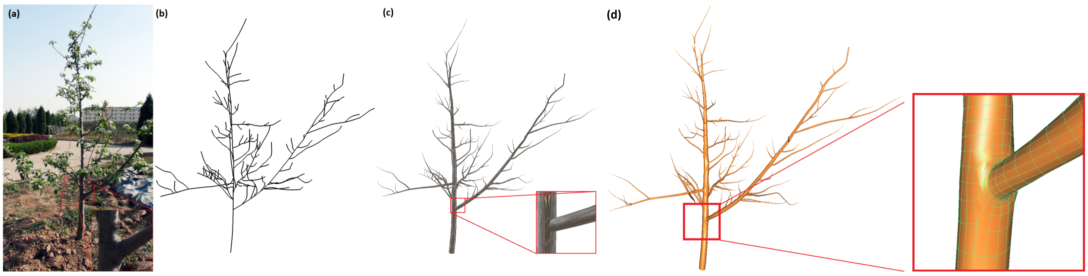


Fig. 1. (d) Prevailing apple tree modeling using our approach. (c) Generalized cylinders represent tree geometries with (b) their skeletons interpolated by Hermite splines. In comparison with a real tree, (a) the generated tree model is a manifold quad-mesh that takes the continuity of branch ramifications into consideration, inherently compensating for the disadvantages of generalized cylinder modeling.

In computer graphics, tree modeling and animation have wide applications in the fields of film production, video games, and virtual reality because plants increase the realism of virtual scenery. In the past three decades, many approaches have been presented to achieve the realistic modeling and animation

[*] Corresponding author. Email: hsj@nwsuaf.edu.cn. College of Information Engineering, Northwest A&F University, No. 22 Xinong Road, Yangling, China. Postal code 712100.

of trees. Tree modeling methods can be categorized into rule-based methods [15,25,27,28], sketch-based methods [21], image-based method [12,30], and modeling methods from point clouds [11,16]. However, most of these studies concentrated on modeling the global morphology of trees, including the complex branch structures and botanical arrangement of leaves. A typical simplified structure to represent tree branches is a generalized cylinder. Many studies in the animation of tree growth [23] and swaying in wind fields [9,10,15,24] also use the same simplified structure to represent bending branch joints.

Although generalized cylinders are efficient and superior in terms of realism in tree modeling, the discontinuity between branches for generalized cylinder representation is obvious, as shown in the close-up in Fig. 1(c). This discontinuity of branches is easy to solve if implicit surface representation is adopted, but this is difficult to control interactively.

The motivation of our work is to create a complete manifold quadrilateral mesh (quad-mesh) effectively, as shown in Fig. 1(d), from user-defined skeletons of a tree (Fig. 1(b)) using Catmull–Clark subdivision for continuous ramification construction, thereby overcoming the drawbacks of both generalized cylinders and implicit surfaces. This study makes two main contributions:

- We propose a method to generate a tree model effectively with smooth ramifications that combines subdivision surface construction with parametric surface construction.
- We propose an optimization algorithm for a tree's control mesh construction using user-defined skeletons.

2 Related Work

2.1 Tree modeling

The earliest developed plant models were procedural models, which generate content using a procedure that has the function of database amplification and can be used to model, for example, plants, buildings, urban environment, and texture. In the case of plant modeling, they have been applied to simulate botanical organs, the growing process, and various plant structures. self-organizing parameter characteristics became the basis of the L-system [26] and self-organizing tree modeling methods [22,32], and we use those characteristics in our approach. Although plausible models have been obtained for the above methods, the final shape of the plant is not easy to control, and many parameters are complex for users to adjust.

A broad trend in computer graphics is data-driven synthesis, where models are created based on real-world measurements, such as those in images or laser scans of geometry. Tan et al. [30] proposed a method of combining input images with user interaction in the construction of trees. Hu et al. [12] modeled trees based on two images from different views with polar constraints for animation using a physical model. Livny et al. [17] reconstructed multiple overlapping trees from point clouds simultaneously without pre-segmentation by applying a series of global optimization-based biologically derived heuristics. Such methods can be extremely effective; however, they are often intended for geometric reconstruction. Hence, maintaining the continuity of ramifications is beyond their scope.

Implicit surface tree modeling is other popular method, which began with an idea presented by Bloomenthal [3]. Compared with parametric surfaces, implicit surfaces are difficult to control and time-consuming, but owing well continuity, noise-resistant, performing Boolean operation easily. There are many types of implicit surfaces, including the convolution surface, which is defined as an iso-surface in a scalar field that convolves a geometric skeleton using a kernel function [4]. An interesting application of the convolution surface is modeling sketch-based models [1, 29, 38], which takes advantage of the rotundity and smoothness of convolution surfaces, which is suitable for tree branches. Another type of typical implicit surface is the Poisson surface [14], which concludes surface reconstruction using Poisson's equation.

There has also been some effort to model smooth joint structures on both parametric surfaces and implicit surfaces for trees (not for botanical trees only). [31] combined generalized subdivision with mesh-based parameterized L-systems to generate smooth ramification structures. [7] generated topologically correct surfaces of branching tubular structures for a vessel tree using the maximal-disc interpolation method. [8] built implicit surfaces as hierarchical BlobTrees [35] and combined surface components in both smooth and non-smooth configurations. [2] proposed an interactive method to refine the joint shape using a user-defined sketch.

Zhu et al. [36] proposed a method of modeling high-quality quad-only tree shapes efficiently based on local convolution surface approximation, which offer credible for our idea of modeling manifold trees with smooth ramifications. However, they focused on remeshing a given triangle mesh of a tree into a quad-mesh. Another solution for continuous vascular structure reconstruction was proposed by [34]. The resulting meshes were not manifold because of the bifurcation tiling scheme in their method, whereas our meshes are manifold because we adopt the cut-paste process.

Although the works of [7] and [34] are similar to our work, the ramifications of botanical trees we attempt to reconstruct possess their own features.

2.2 Subdivision

In this section, we show why the Catmull–Clark scheme [5] appears to be a better choice for our application than other schemes by comparing results on a cuboid-like mesh. The resulting surfaces of the Loop [18], Catmull–Clark, Butterfly [20], and Doo–Sabin [6] schemes obtained by subdividing a cube are shown in Fig. 2. The figure shows that the Butterfly scheme increased irregularly based on the original mesh, and it was difficult for it to satisfy the visual requirements. The Loop and Doo–Sabin schemes maintained the original shape to some degree, which was subtly affected at the edges of the cube. The result of the Catmull–Clark scheme is the closest to a sphere, with little influence on the weights of the edges from the initial mesh. The surface of tree branches can be easily approximated based on a cuboid-like initial mesh for Catmull–Clark subdivision. Thus, we adopt a classical Catmull–Clark scheme for tree ramification representation.

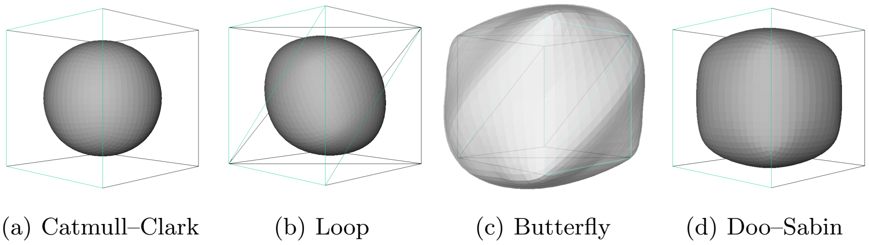

(a) Catmull–Clark (b) Loop (c) Butterfly (d) Doo–Sabin

Fig. 2. Results of applying various subdivision schemes to a cube. Because Loop and Butterfly are triangular schemes, the cube was triangulated first before the subdivision scheme was applied.

Catmull–Clark scheme subdivision is based on the tensor product bicubic spline and generates surfaces that are C^2 everywhere except at extraordinary vertices, where they are C^1.

It is not difficult to conclude that any arbitrary polygonal mesh can be reduced to a quad-mesh once the first Catmull–Clark subdivision is executed. Furthermore, the number of extraordinary vertices is determined simultaneously and tends to be unchanged after one or more Catmull–Clark subdivisions. The remaining problem is how to generate an initial mesh to connect these branches appropriately with little visual distortion, which is introduced in the following section.

3 Overview

The final aim of our work is to create a manifold tree quad-mesh with tolerable visual distortion and smooth ramifications, which is described by the following objective function:

$$f(Rems) = \sum_{i=1}^{N} RU(Rems_i) + \sum RC(Rem_{i,j}) + \sum CP(Rem_{i,k}), \tag{1}$$

where *Rems* denotes the set of all ramifications in a tree extracted from the skeleton information of branches, and is regarded as an independent variable in the objective function. Sub-objective function *RU* is the distortion function of the ramification unit for Catmull–Clark subdivision. $Rems_i$ is the i-th ramification in the set $Rems$. Sub-objective function RC is the distortion function of the ramification connection between two neighbor ramifications. $Rem_{i,j}$ is the connection between the i-th ramification and j-th ramification. Sub-objective function CP (cut-paste) is the fitness function of the pasted ramification between two overlapping ramifications. $Rem_{i,k}$ is the i-th ramification merged with the k-th ramification.

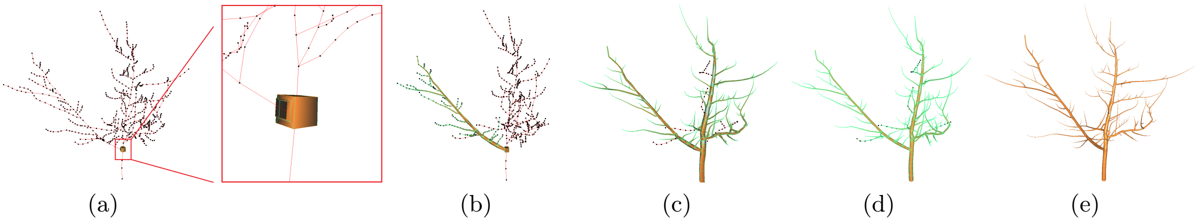


Fig. 3. Overview of our tree modeling approach. (a) Users define the skeleton information of branches. Extract the close-up of one ramification skeleton and create a basic ramification unit. (b) Propagate ramifications for a single ramification unit. (c) Connect ramification units using an optimization algorithm and distinguish the additional ramifications from the main mesh simultaneously. (d) First, subdivide the main mesh to which the additional ramifications are pasted back. (e) Obtain the final mesh with one more subdivision.

Hence, we convert the problem into creating a tree model with minimum $f(Rems)$. Fig. 3 shows the workflow of our tree modeling system, which consists of three parts.

First, skeletons of a tree are defined by the user. Then, basic ramification units, which can represent continuous ramification structures after Catmull–Clark subdivision, are created with an RU value equal to zero as the initial state using the skeleton information of branches in the local coordinate system, thereby setting their parent node as the origin. The ramification units are mapped back to the world coordinate system. This part of the procedure is described in Section 4.

Once all ramifications are arranged, they can be easily sorted according to the order of their parent nodes and checked to determine whether they are connectable ramifications or additional ramifications. A bottom-up optimization algorithm is applied recursively to adjust all connections among connectable ramifications, thereby striking a balance between the RU and RC functions in addition to making the ramification propagate. We obtain the main mesh of the tree at the end of this step. The details of this part of the procedure are described in Section 5.

If there exist any additional ramifications, then this means that some child branches that could destroy the manifold of the tree mesh have been detected. They should be cut and pasted into the main mesh after its first Catmull–Clark subdivision. The CP function indicates the distortion in this operation as discussed in Section 6.

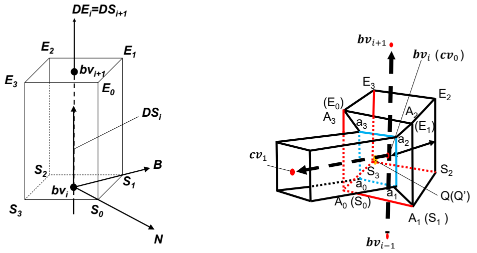
4 Basic Branch Unit Creation

Fig. 4 shows two typical basic ramification units: the connection unit shown in Fig. 4(a) is for those segments of branches that do not have any child branches, whereas the ramification unit in Fig. 4(b) is for a child branch that has start direction $(\boldsymbol{cv}_1 - \boldsymbol{cv}_0)$ located in the i-th node $\boldsymbol{bv}_i$ of its parent branch, counted from the root node.

For the connection unit between $\boldsymbol{bv}_i$ and $\boldsymbol{bv}_{i+1}$, $\boldsymbol{DS}_i$ is the unit start direction that is the same as $(bv_{i+1} - bv_i)$, and $\boldsymbol{DE}_i$ is the unit end direction. Additionally, $\boldsymbol{S}_j$(j=0,1,2,3) denotes the vertices of the start boundary and $\boldsymbol{E}_j$ denotes the vertices of the end boundary. They are created by basis $\{\boldsymbol{B}, \boldsymbol{N}, \boldsymbol{DS}_i\}$. Both the start and end boundaries of ramifications are sorted clockwise when these ramifications are created.

The ramification unit is an expansion of the connection unit, in addition to a sub-branch. $\boldsymbol{Q}$ is the intersection of main face $\boldsymbol{ABCD}$ and the child branch skeleton. Subface $\boldsymbol{abcd}$ is also called a sub-branch start boundary, which is recorded for boundary calculation using Catmull–Clark subdivision as explained in Section 6. At this step, $\boldsymbol{Q}$ is also the center of main face $\boldsymbol{A}_0\boldsymbol{A}_1\boldsymbol{A}_2\boldsymbol{A}_3$, and $\boldsymbol{Q}'$, which superposes $\boldsymbol{Q}$, is the center of subface $\boldsymbol{a}_0\boldsymbol{a}_1\boldsymbol{a}_2\boldsymbol{a}_3$. The sum of cosine distances between the pairs of vectors is selected as the RU function, that is,

$$RU(Rems_i) = \sum_{k=0}^{3} CosDistance(\boldsymbol{QA}_k, \boldsymbol{Q'a}_k) = \sum_{k=0}^{3} \frac{\boldsymbol{QA}_k \cdot \boldsymbol{Q'a}_k}{\|\boldsymbol{QA}_k\| \cdot \|\boldsymbol{Q'a}_k\|}. \tag{2}$$



(a) Connection unit (b) Ramification unit

Fig. 4. Two typical basic branch units created according the skeleton

4.1 Criteria

A suitable basic ramification unit plays a large role in ramification representation. Taking the properties of Catmull–Clark subdivision into consideration, the following criteria should be satisfied naturally for high-quality tree quad-mesh construction.

1. A ramification unit is created corresponding to a subbranch.

2. Each RU value of the ramification unit when it is built is zero (the minimum value) initially because this value will be increased in the subsequent connection optimization step, so a zero value simplifies the calculation.

3. The diameter of a branch should be multiplied by correction factor α to counteract the shrinkage caused by Catmull–Clark subdivision (particularly the first two subdivisions).

4.2 Ramification Unit in the Local Coordinate System

Let $\{\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}\}$ denote the basis of the world coordinate system in $\boldsymbol{R}^3$, in which the skeletons of all branches from a tree are user-defined. Local coordinate systems with basis $\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}$ can be built for each child branch, and their mapping to the world coordinate system is decomposed into one 1×3 translation vector $\boldsymbol{T}_0$, and two 3×3 rotation matrices $\boldsymbol{Rot1}$ and $\boldsymbol{Rot2}$. Given a vector in the world coordinate system, $\boldsymbol{P}$, and a vector in a local coordinate system, $\boldsymbol{p}$, we have the following surjective mapping equations:

$$\boldsymbol{p} = \boldsymbol{Rot2} \cdot \boldsymbol{Rot1} \cdot (\boldsymbol{P} - \boldsymbol{T}_0) \tag{3}$$

$$\boldsymbol{P} = \boldsymbol{Rot1}^T \cdot \boldsymbol{Rot2}^T \cdot \boldsymbol{p} + \boldsymbol{T}_0, \tag{4}$$

where $\boldsymbol{T}_0$ is $\boldsymbol{bv}_i$ in Fig. 4; $\boldsymbol{Rot1}$ is the rotation matrix calculated by the Rodrigues rotation formula to make the parent branch direction $(\boldsymbol{bv}_{i+1} - \boldsymbol{bv}_i)$ aligned to the $\boldsymbol{y}$ axis, whereas $\boldsymbol{Rot2}$ makes $\boldsymbol{y} \times (\boldsymbol{cv}_1 - \boldsymbol{cv}_0)$ aligned to $\boldsymbol{x}$. Then ramification unit Rem_i, such as that in Fig. 4(b), contains set of vertices V_{local}, and set of faces F is built in this local coordinate system. Then, according to Eq. 4, Rem_i in the world coordinate system can be obtained easily by translating V_{local} to V, which is a corresponding set of vertices in the world coordinate system. With the help of the local coordinate system, it becomes trivial to check whether the neighbor ramifications are connectable by converting them into the same local coordinate system and projecting them into the same plane.

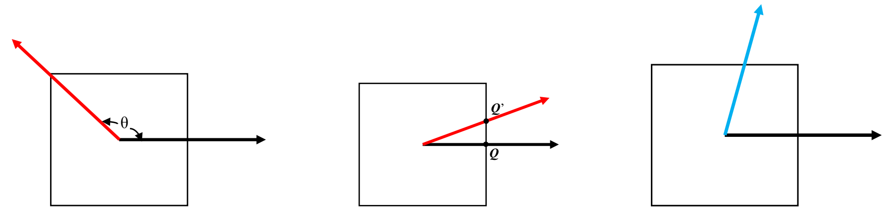
5 Ramification Unit Connection Optimization and Propagation

After all ramifications are built and classified, we link all connectable ramifications first using a connection optimization algorithm to alleviate distinct distortion among ramifications. Thus, we should first distinguish connectable ramifications from additional ramifications.

If a node has more than one child branch, then one child branch is selected to create a ramification as the "current ramification", and the other branches are checked to determine whether they are additional ramifications, as shown in Fig. 5. According to the shape of the ramification unit, there are two cases in which a child branch can be considered as an additional ramification:

1. The absolute value of the angle between two projected vectors of the child branch in the current ramification modulo 45° is smaller than 10° (Fig. 5(a)).

2. Q and Q', which are the intersection points of two child branches on the current ramification, are not on the same face (Fig. 5(b)).



(a) $|(\theta)mod(45°)| < 10°$ (case 1)

(b) Q and Q' are in the same face (case 2)

(c) Connectable ramification

Fig. 5. Distinguishing the ramification type in the transverse view of the ramification unit cross section (black square). The black arrow is the projected vector of the child branch in the current ramification. The red arrow is that in the additional ramification. The blue arrow is that in the connectable ramification, which can be attached to the current ramification trivially before the subdivision step.

In this section, sub-objective functions RU and RC must be considered simultaneously. Thus, f_2, which is the objective function in this step, is

$$f_2(Rems) = \sum_{i=1}^{N} RU(Rems_i) + \sum RC(Rem_{i,j}). \tag{5}$$

To determine the minimum value, we divide this task into two independent parts: a radial neighbor ramification connection and axial connection calibration with repulsion equilibrium.

The radial neighbor ramification connection attempts to determine corresponding adjacent start and end boundaries between neighbor ramifications along the directions of skeletons:

$$Idx = argmin \sum_{k=0}^{m-1} CosDistance(CE_k^{(i)}, CS_{(k+Idx)mod(m)}^{(i+1)}), \tag{6}$$

where $CE_k^{(i)}$ is the end boundary in Rem_i and $CS_k^{(i+1)}$ is the start boundary in Rem_{i+1}. When the ramifications on one branch connect correctly, axial connection calibration is applied to expand the connection space between neighbor ramifications, which is described as

$$\begin{cases} S_i^{(k)} = argmax(\sum_{i=0}^{3} \left\| E_i^{(k-1)} - S_i^{(k)} \right\|), s.t. \left\| Q^{(k)} - S_i^{(k)} \right\| \geq \alpha \cdot Dia_{max} \\ E_i^{(k)} = argmax(\sum_{i=0}^{3} \left\| E_i^{(k)} - S_i^{(k+1)} \right\|), s.t. \left\| E_i^{(k)} - Q^{(k)} \right\| \geq \alpha \cdot Dia_{max} \end{cases} \tag{7}$$

For a pair of neighbor ramifications, the solution of Eq. 7 can be explained by Fig. 6. Δx_e is the axial movement of the end boundary in Rem_i, and based on the law of cosines, the increment of RU is

$$\Delta RU = 4 \cdot cos(\angle A_2 Q_i A_2') = 2\frac{\|Q_i A_2\|^2 + \|Q_i A_2'\|^2 - \|A_2 A_2'\|^2}{\|Q_i A_2\| \cdot \|Q_i A_2'\|} \tag{8}$$

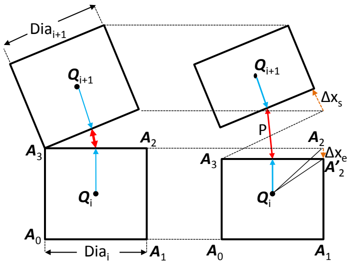


Fig. 6. Axial equilibrium between two connectable ramifications

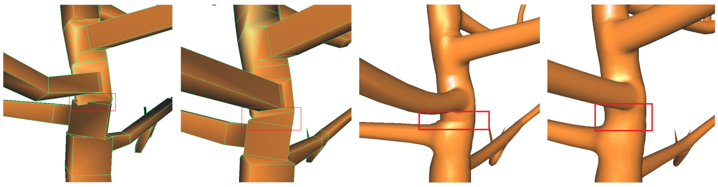
where Δx_e equals $\|A_2A_2'\|$. $\|Q_iA_2\|$ is the radius of the sphere tangent to edges of the cube with an edge length of Dia_i before axial connection calibration. The constant 4 results from the symmetry of the cube. Simultaneously, the increment of RC is

$$\Delta RC = \begin{cases} -(1-\frac{P}{T}) \cdot CosDistance(CE_k^{(i)}, CS_{(k+Idx) mod(m)}^{(i+1)}) & 0 \le P < T \\ 0 & P \ge T \end{cases}, \quad (9)$$

where P is the distance between the center of the end boundary in Rem_i and that of the start boundary in Rem_{i+1}, which can also be regarded as the repulsion between the neighbor ramifications because ΔRC is only valid for a sufficiently close distance. Threshold T is set to Dia_i. When ΔRC is negative, this means that axial movement Δx_e is helpful for reducing the visual distortion of the connection.

The solution retrieval of Δx_s and Δx_e for $min(\Delta RU + \Delta RC)$ is conducted iteratively, and the effect of connection optimization is shown in Fig. 7. The figure shows that connection optimization avoids the overlap between the neighbor ramifications (Fig. 7 (a)). Moreover, the subdivision surface of the ramification connection (in the red box) after optimization (Fig. 7(d)) is smoother than that without optimization (Fig. 7 (c)).

By contrast, for any Δx_e, if $min(\Delta RU + \Delta RC)$ is always larger than given additional ramification threshold T_{add}, then Rem_i is considered as an additional ramification and should not be connected into main mesh in this step; it is also the last case to obtain an additional ramification.



(a) Original mesh (b) Connection optimization (c) Subdivision for (a) (d) Subdivision for (b)

Fig. 7. Effect of connection optimization

After all additional ramifications for the next step have been picked up, we also obtain connectable ramification set $Rams_c$. Then, the connection optimization algorithm is applied from bottom to top in $Rams_c$ recursively for sub-branch propagation, as shown in Fig. 3(b), which can be described by the propagation Algorithm 1. The main idea of this algorithm is to search all child branches along a branch's node list (skeleton) and connect corresponding connectable ramifications successively. When this algorithm is applied from the root node of a tree, we can obtain the main mesh of the tree, and the remaining task is a cut-paste operation for additional ramifications.

Algorithm 1 Sub-branch propagation

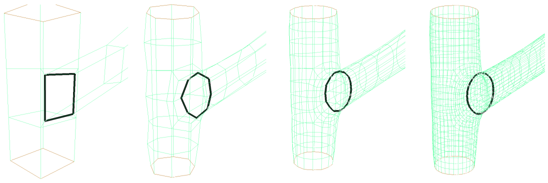
```
Input: Current ramification that needs to grow up $CurrentRam$,
       connectable ramification set $Rams_c$,
       all node lists' (skeletons') set $Branches$;
Output: Grown up ramifications $CurrentRam'$;
Define an empty sub-branch's ramification queue, $qRam$;
Define array $Ram\_array$ to record the current branch's child branch identifier (ID) using its node list with initial value -1;
Define $CurrentBranch = Branches[CurrentRam.ChildNodeID]$;
Define current child ID $CurrentCid = 0$;
for $i > CurrentBranch.Cid.size$ do
    $SubBranch = Branches[CurrentBranch.Cid[CurrentCid]]$;
    $Ram\_array[SubBranch.sid] = SubBranch.Cid[CurrentCid]$;
    i=i+1;
end for
$CurrentCid = 0$;
while $j < CurrentBranch.NodeList.size$ do
    if $Ram\_array[CurrentCid] \neq -1$ then
        $SubBranch = Branches[CurrentBranch.Cid[CurrentCid]]$;
        Define temporary ramification
        $Ram = Rams_c[SubBranch.RamID]$;
        Run this algorithm recursively for $Ram$ and obtain grown up ramification $RamG$;
        $qRam$.push($RamG$);
    end if
end while
Define temporary ramification $Ram'$
while $iRam.size > 0$ do
    $Ram' = iRam$.pop()
    if $iRam.size > 0$ then
        Define the next ramification that needs to be connected, $Ram2 = iRam$.pop()
        Connect $Ram'$ with $Ram2$ according to Eq. 5 and obtain new $Ram'$ ;
        Update the end boundary indices of $Ram'$;
    else
        Create a connection unit along the remainder of CurrentBranch.NodeList
    end if
end while
return $Ram'$.
```

6 Additional Ramification Cut-Paste

As all basic ramifications are constructed first, our modeling method has a local priority. Distortion accumulates if all basic ramifications are connected, and the two-manifold structure of the tree modeling surface is distorted by the overlap of ramifications. If a ramification can cause high distortion or overlaps with other ramifications, then we select it as an additional ramification before the ramification connection optimization step to avoid it having a bad effect on the entire tree. To merge those additional ramifications back into the main mesh created in the ramification connection optimization step, cut-paste is performed.

Only when additional ramifications exist can this operation be implemented to merge those additional ramifications into the main mesh after the first Catmull–Clark subdivision. Each additional ramification grows up according to Algorithm1, and is cut alone with the sub-branch boundary after the first subdivision. Then, according to the Catmull–Clark subdivision process (Section 2.2), the original sub-branch boundary is determined by recording the new edge vertices that were generated from vertices that belong to the original sub-branch boundary. This method stably calculates the current sub-branch boundary shown in Fig. 8. As the sub-branch boundary is known, we can extract the vertex set and corresponding face set of the grown sub-branch beginning with any seed vertex in this sub-branch.

Then, a segment-quad-face intersection test based on segment-triangle one [19] is implemented to determine the intersection face in the main mesh in addition to the closest vertex. To save time, we limit the intersection test scope to a sphere, with the joint node of the additional ramification as the center point and 1.5 times its diameter as the radius. The one-ring neighbor of the closest vertex constitute the



(a) Original boundary (b) 1st subdivision (c) 2nd subdivision (d) 3rd subdivision

Fig. 8. Boundary calculation after Catmull–Clark subdivision

paste-destination boundary. The sub-branch boundary can match the paste destination boundary using an equation similar to Eq. 6:

$$Idx = argmin \sum_{k=0}^{m-1} CosDistance(\boldsymbol{SB}_k, \boldsymbol{PB}_{(k+Idx)mod(m)}), \tag{10}$$

where $\boldsymbol{SB}$ and $\boldsymbol{PB}$ are the sub-branch boundary and paste destination boundary, respectively, which were projected into same plane following their center point alignment. Additionally, $m = 8$, in this case.

When merging the two boundaries, sub-objective function CP between Rem_i and Rem_k is described as

$$CP(Rem_{i,k}) = \beta \sum_{j=0}^{m-1} CosDistance(\boldsymbol{SB}_j^{(i)}, \boldsymbol{PB}_j^{(k)}) + (1-\beta) \sum_{j=0}^{m-1} \|\boldsymbol{SB}_j^{(i)} - \boldsymbol{PB}_j^{(k)}\|, \tag{11}$$

where β is a weight factor for the boundary merge, $\boldsymbol{SB}^{(i)}$ is the sub-branch boundary in Rem_i, and $\boldsymbol{PB}^{(k)}$ is the paste destination boundary close to Rem_k, which is regarded as an additional ramification. As we can see, two factors contribute to cut-paste distortion: the angle deflection between $\boldsymbol{PB}$ and $\boldsymbol{SB}$, and the Euclidean distance between them. Thus, we need to rotate and translate $\boldsymbol{PB}$ and $\boldsymbol{SB}$ to determine the minimum CP. Fig. 9(a) to (d) show the entire cut-paste process as an example. The additional ramifications' cut-paste process is described as follows:

1. Calculate the intersection point between the skeleton in an additional ramification and main mesh after the first Catmull–Clark subdivision.
2. Determine the closest vertex to the intersection point in the main mesh.
3. Delete the face that contains the closest vertex and its one-ring neighbor in the main mesh, and obtain the entire boundary as a paste destination boundary.
4. Extract a sub-branch from the additional ramification along with its sub-branch boundary.
5. Merge the sub-branch boundary and paste the destination boundary with minimizing Eq. 11 using the rotation and translation operation.
6. After all additional ramifications are cut and pasted, apply the Catmull–Clark subdivision to obtain the final modeling of the tree.

Fig. 9(e) shows the generality of our cut-paste process by pasting another additional ramification into the main mesh with a different position, rotation, and diameter.

7 Results and discussion

In this section, we present the results of our method using a sketching tree modeling interface. To obtain the 3D skeleton of branches for our method, we drew and adjusted our tree from both the front view and side view, adopting the same method as that in [12]. The main user interfaces of the tree modeling system are shown in Fig. 10(a) and (b), which denote two 2D views. We input two pictures of a tree with its camera parameters and sketched the 2D skeletons along the pictures so that the 3D skeletons of branches could be calculated.

In our first experiment, as Fig. 10 shows, we attempted to reconstruct a simple binary tree whose point cloud was obtained using structure from motion as ground truth from photographs that covered the tree fork 360°.

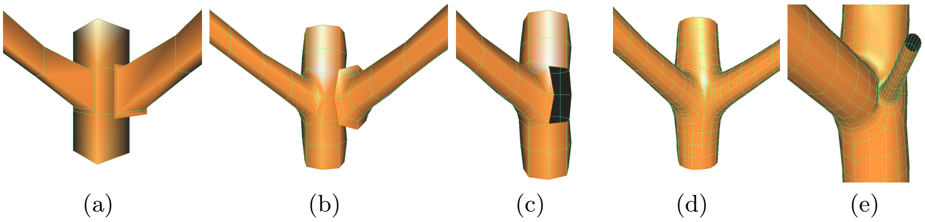
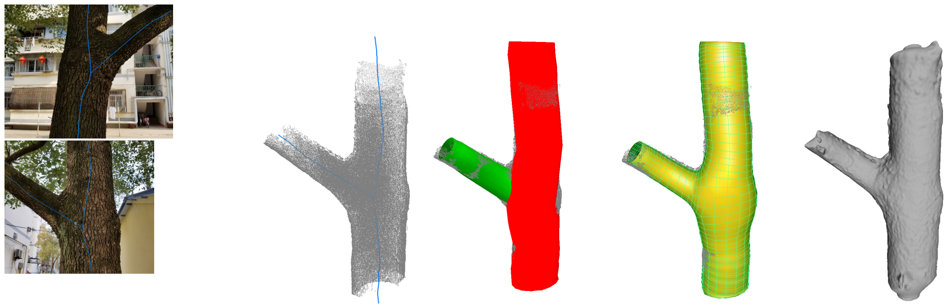


Fig. 9. Cut-paste process for an additional ramification. (a) Merge the sub-branch at the right-hand side, which is an additional ramification, with the main mesh. (b) Extract the additional ramification with its sub-branch boundary after the first Catmull–Clark subdivision. (c) Determine the closest vertex in the main mesh in addition to its one-ring neighbor as the paste-destination boundary, which is used to delete the overlapping face when implementing the paste operation. (d) Match the sub-branch boundary of the additional ramification to the paste-destination boundary and merge the two branches using one more subdivision. (e) Another ramifications' cut-paste result for an additional branch with a different diameter, rotation, and position.

Compared with the classical generalized cylinder method, the results in Fig. 10(d) and (e) demonstrate that our approach modeled tree ramifications as a manifold, preserving the shape and features expressed by generalized cylinders faithfully. The gray parts of the point cloud indicate the difference between the modeling surface and real surface. Without any special approximate algorithm to fit the point cloud, our result in Fig. 10(e) had fewer gray parts than that for the generalized cylinders, particularly around the ramification part, which means that our method was more suitable for describing the tree structure than generalized cylinders.

This experiment also demonstrated that our result, which was available by drawing simple skeletons, was a suitable summary and simplification of that created by screened Poisson surface reconstruction [13] from a dense point cloud. Although the Poisson surface had more details, our result demonstrated a similar global geometric impression to its result. By contrast, the Poisson surface could not express the texture of bark well geometrically because it was limited by the density of the point cloud obtained from pictures. In this case, texture mapping for a bump map may be a better choice to represent tree bark, and our surface could be smoother and easily parameterized for texture mapping.



(a) Real picture in two views (b) Point cloud (c) Generalized cylinders (d) Our approach (e) Poisson surface

Fig. 10. Examples of a model for a ramification of a real-tree

Another experiment was conducted to verify whether the cut-paste step was suitable for multi-furcation ramifications. In this experiment, we set all the ramifications as additional ones manually for verification. We present the results for ramifications with a furcation number from 4 to 7 (additional ramification numbers were 3 to 6) in Fig. 11. The two-manifold property for the meshes were maintained well as the furcations increased, which means that the cut-paste process that we adopted decreased distortion and avoided overlap for multi-ramifications . Tab. 1 records the face number for the final mesh, and the time cost for cut-paste and subdivision; the furcation number is in proportion to all of other items.

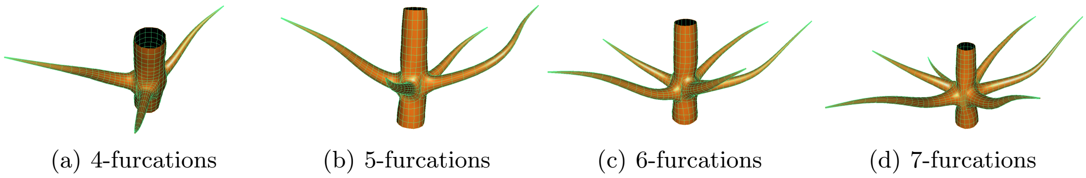

(a) 4-furcations (b) 5-furcations (c) 6-furcations (d) 7-furcations

Fig. 11. Multi-furcation ramification construction

Table 1. Time cost for multi-furcation ramification construction

Furcation number	Face number	Cut-Paste(ms)	Twice Subdivison(ms)
4	1104	82	41
5	1408	121	49
6	1904	179	76
7	2208	259	98

As the results of the experiments have demonstrated, our method was suitable for tree ramification modeling. Our method was applied in the next experiment to some complete trees. Fig. 12 shows a variety of results generated from skeletons of different trees. The complexity of the skeletons ranged from a small number to a large number. All the meshes of trees were manifold, with continuous ramifications, which increased the realism of the trees. Our method obtained a complete tree model for different types of trees, and maintained the two-manifold property of their ramifications.

The corresponding time cost of these trees in each step is shown in Tab. 2. The additional ramifications were recognized automatically according the section 5. The table shows that, in our method, a complete tree was modeled in a short time. Taking Tab. 1 into consideration, we can found that the average time consumed in the cut-paste step per additional ramification was far longer than that in Tab. 2. Thus, this also demonstrated that the most time-consuming step was cut-paste because of the intersection test between additional ramifications and the main mesh, which is why we limited the search scope in this step. Further research is necessary to reduce the time of the cut-paste step.

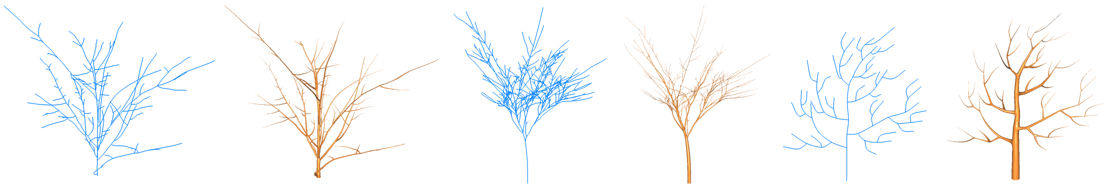


Fig. 12. Variety of results generated from user-defined skeletons (blue lines) of different trees. From left to right: cherry tree (Tree1), maple tree (Tree2), and manual apple tree (Tree3).

8 Conclusions

We proposed an effective and intuitive tree modeling system to generate manifold quad-meshes with smooth and continuous ramification structures. The resulting surface was generated using a Catmull–Clark subdivision scheme directly without any extra virtualization algorithm. To improve the surface quality of the tree and retain the two-manifold property of the mesh, ramification connection optimization and additional ramification cut-paste were conducted for our local priority mesh generation algorithm.

The user-defined skeleton information of the branches was intuitive and essential as input, which decreased the difficulty of interactive control for tree modeling. Our resulting meshes were purely quadrilateral with continuous ramifications, which makes them similar to those that adopt generalized cylinders, and can be a reasonable summary of the Poisson surface in a short time.

Table 2. Time cost in each step of modeling for different trees

Tree No.	Branches number	Connection Optimization(ms)	Additional ramifications number	Cut-Paste(ms)	1st Subdivison (ms)	2nd Subdivison (ms)
Tree1	85	1572	5	4991	361	3601
Tree2	200	2101	4	8486	547	4632
Tree3	42	242	0	0	194	414

References

1. Alexe, I.A., Barthe, L., Cani, M., Gaildrat, V.: Shape modeling by sketching using convolution surfaces. In: 34. International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2007, San Diego, California, USA, August 5-9, 2007, Courses. p. 39 (2007)
2. Angles, B., Tarini, M., Wyvill, B., Barthe, L., Tagliasacchi, A.: Sketch-based implicit blending. ACM Trans. Graph. **36**(6), 181:1–181:13 (2017)
3. Bloomenthal, J.: Skeletal design of natural forms. Computer Science University of Calgary (1995)
4. Bloomenthal, J., Shoemake, K.: Convolution surfaces. In: Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1991, Providence, RI, USA, April 27-30, 1991. pp. 251–256 (1991)
5. Catmull, E., Clark, J.: Recursively generated b-spline surfaces on arbitrary topological meshes. Computer-Aided Design **10**(6), 350 – 355 (1978)
6. Doo, D., Sabin, M.: Behaviour of recursive division surfaces near extraordinary points. Computer-Aided Design **10**(6), 356–360 (1978)
7. Felkel, P., Wegenkittl, R., Buhler, K.: Surface models of tube trees. In: 2004 Computer Graphics International Conference (CGI 2004). pp. 70–77. Crete, Greece (1 2004)
8. Galbraith, C., Mündermann, L., Wyvill, B.: Implicit visualization and inverse modeling of growing trees. Comput. Graph. Forum **23**(3), 351–360 (2004)
9. Habel, R., Kusternig, A., Wimmer, M.: Physically guided animation of trees. Comput. Graph. Forum **28**(2), 523–532 (2009)
10. Hu, S., Chiba, N., He, D.: Realistic animation of interactive trees. The Visual Computer **28**(6-8), 859–868 (2012)
11. Hu, S., Li, Z., Zhang, Z., He, D., Wimmer, M.: Efficient tree modeling from airborne lidar point clouds. Computers & Graphics **67**, 1–13 (2017)
12. Hu, S., Zhang, Z., Xie, H., Igarashi, T.: Data-driven modeling and animation of outdoor trees through interactive approach. The Visual Computer **33**(6), 1017–1027 (2017)
13. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. Acm Transactions on Graphics **32**(3), 1–13 (2013)
14. Kazhdan, M.M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, Cagliari, Sardinia, Italy, June 26-28, 2006. pp. 61–70 (2006)
15. Li, C., Deussen, O., Song, Y.Z., Willis, P., Hall, P.: Modeling and generating moving trees from video. ACM Trans. Graph. **30**(6), 127 (2011)
16. Livny, Y., Pirk, S., Cheng, Z., Yan, F., Deussen, O., Cohen-Or, D., Chen, B.: Texture-lobes for tree modelling. ACM Trans. Graph. **30**(4), 53:1–10 (2011)
17. Livny, Y., Yan, F., Olson, M., Chen, B., Zhang, H., El-Sana, J.: Automatic reconstruction of tree skeletal structures from point clouds. ACM Trans. Graph. **29**(6), 151:1–151:8 (2010)
18. Loop, C.: Smooth subdivision surfaces based on triangles (01 1987)
19. Möller, T., Trumbore, B.: Fast, minimum storage ray/triangle intersection. In: 32. International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2005, Los Angeles, California, USA, July 31 - August 4, 2005, Courses. p. 7 (2005)
20. Nira, D., David, L., John, A., G.: A butterfly subdivision scheme for surface interpolation with tension control. ACM Trans. Graph. **9**(2), 160–169 (1990)
21. Okabe, M., Owada, S., Igarashi, T.: Interactive design of botanical trees using freehand sketches and example-based editing. Computer Graphics Forum **24**(3), 487–496 (2005)
22. Palubicki, W., Horel, K., Longay, S., Runions, A., Lane, B.: Self-organizing tree models for image synthesis. Acm Transactions on Graphics **28**(3), 1–10 (2009)
23. Pirk, S., Niese, T., Deussen, O., Neubert, B.: Capturing and animating the morphogenesis of polygonal tree models. ACM Trans. Graph. **31**(6), 169:1–10 (2012)
24. Pirk, S., Niese, T., Hädrich, T., Benes, B., Deussen, O.: Windy trees: Computing stress response for developmental tree models. ACM Trans. Graph. **33**(6), 1–11 (Nov 2014)

25. Pirk, S., Stava, O., Kratt, J., Said, M.A.M., Neubert, B., Měch, R., Benes, B., Deussen, O.: Plastic trees: Interactive self-adapting botanical tree models. ACM Trans. Graph. **31**(4), 50:1–10 (2012)
26. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants (1990)
27. Runions, A., Lane, B., Prusinkiewicz, P.: Modeling trees with a space colonization algorithm. In: Proceedings of the Third Eurographics Conference on Natural Phenomena. pp. 63–70. NPH'07, Eurographics Association, Aire-la-Ville, Switzerland (2007)
28. Stava, O., Pirk, S., Kratt, J., Chen, B., Mech, R., Deussen, O., Benes, B.: Inverse procedural modelling of trees. Computer Graphics Forum **33**(6), 118–131 (2014)
29. Tai, C., Zhang, H., Fong, J.C.: Prototype modeling from sketched silhouettes based on convolution surfaces. Comput. Graph. Forum **23**(1), 71–84 (2004)
30. Tan, P., Zeng, G., Wang, J., Kang, S.B., Quan, L.: Image-based tree modeling. ACM Trans. Graph. **26**(3), 87 (2007)
31. Tobler, R.F., Maierhofer, S., Wilkie, A.: Mesh-based parametrized l-systems and generalized subdivision for generating complex geometry. International Journal of Shape Modeling **8**(2), 173–191 (2002)
32. Wang, Y., Xue, X., Jin, X., Deng, Z.: Creative virtual tree modeling through hierarchical topology-preserving blending. IEEE Trans Vis Comput Graph **1**(99), 2521–2534 (2017)
33. Wu, X., Luboz, V., Krissian, K., Cotin, S., Dawson, S.: Segmentation and reconstruction of vascular structures for 3d real-time simulation. Medical Image Analysis **15**(1), 22–34 (2011)
34. Wyvill, B., Guy, A., Galin, E.: Extending the CSG tree - warping, blending and boolean operations in an implicit surface modeling system. Comput. Graph. Forum **18**(2), 149–158 (1999)
35. Xiaoqiang, Z., Xiaogang, J., Lihua, Y.: High-quality tree structures modelling using local convolution surface approximation. The Visual Computer **31**(1), 69–82 (2015)
36. Zhu, X., Jin, X., Liu, S., Zhao, H.: Analytical solutions for sketch-based convolution surface modeling on the GPU. The Visual Computer **28**(11), 1115–1125 (2012)